IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES
Attorney Docket № 14222US02 (BU 2739)

Electronically filed on 27-OCT-2010

In re Application of:

Martin Lund

Serial No.: 10/647,963

Filing Date: August 26, 2003

For: SYSTEM AND METHOD FOR INTEGRATING MULTISERVER PLATFORMS

Examiner: BARQADLE, YASIN M

Group Art Unit No.: 2456

Confirmation No.: 5243

Customer No.: 23446

APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This is an appeal from an Office Action dated May 11, 2010 ("Final Office Action"), in which claims 1-15 were finally rejected. The Appellant respectfully requests that the Board of Patent Appeals and Interferences ("Board") reverses the final rejection of claims 1-15 of the present application. The Appellant notes that this Appeal Brief is timely filed within the one-month period for reply from the mailing of the Notice of Panel Decision from Pre-Appeal Brief Review that ends on **October 28, 2010**.

REAL PARTY IN INTEREST
(37 C.F.R. § 41.37(c)(1)(i))

Broadcom Corporation, a corporation organized under the laws of the state of California, and having a place of business at 5300 California Avenue, Irvine, California 92617, has acquired the entire right, title and interest in and to the invention, the application, and any and all patents to be obtained therefor, as set forth in the Assignment recorded at Reel 014447, Frame 0015 in the PTO Assignment Search room.

RELATED APPEALS AND INTERFERENCES
(37 C.F.R. § 41.37(c)(1)(ii))

The Appellant is unaware of any related appeals or interferences.

STATUS OF THE CLAIMS
(37 C.F.R. § 41.37(c)(1)(iii))

Claims 1-15 were finally rejected in the Final Office Action mailed May 11, 2010. Pending claims 1-15 are the subject of this appeal.

The present application includes claims 1-15, which are pending in the present application. Claims 1-15 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Publication No. 2002/0188718, by McGraw et al. *See* Final Office Action at pages 4-7.

The Appellant identifies claims 1-15 as the claims that are being appealed. The text of the pending claims is provided in the Claims Appendix.

STATUS OF AMENDMENTS
(37 C.F.R. § 41.37(c)(1)(iv))

The Appellant amended claims 1, 2, 5, 6 and 10 for clarification in the Final Office Action response filed July 6, 2010. The amendment was entered and the rejections were maintained in the Advisory Action mailed July 16, 2010.

SUMMARY OF CLAIMED SUBJECT MATTER
(37 C.F.R. § 41.37(c)(1)(v))

Independent claim 1 recites the following:

A method for communication information in a server platform,[1] the method comprising:

receiving at least one packet from at least a first switch blade[2] associated with a first multiserver platform;[3]

determining at least a server blade[4] associated with a second multiserver platform[5] for receiving at least a portion of said received at least one packet;[6] and

[1] *See* present application, *e.g.*, at page 4, lines 3-4; page 7, lines 3-4.
[2] *See id.*, *e.g.*, at Figure 1 (160); Figure 2 (202); Figure 3 (306); Figure 4 (408); Figure 6 (607).
[3] *See id.*, *e.g.*, at page 4, lines 4-6; page 7, lines 4-5; Figure 1 (100); Figure 2 (201); Figure 3 (303); Figure 4 (402); Figure 6 (604).
[4] *See id.*, *e.g.*, at Figure 4 (426).
[5] *See id.*, *e.g.*, at Figure 3 (304); Figure 4 (422); Figure 6 (605).
[6] *See id.*, *e.g.*, at page 4, lines 6-8; page 7, lines 6-7.

routing said at least a portion of said at least one received packet to at least said server blade.[7]

Claims 2-4 are dependent upon claim 1.

Independent claim 5 recites the following:

A machine-readable storage having stored thereon, a computer program having at least one code section for communicating information in a server platform, the at least one code section being executable by a machine for causing the machine to perform steps[8] comprising:

receiving at least one packet from at least a first switch blade[9] associated with a first multiserver platform;[10]

determining at least a server blade[11] associated with a second multiserver platform[12] for receiving at least a portion of said received at least one packet;[13] and

routing said at least a portion of said at least one received packet to at least said server blade.[14]

[7] *See* present application, *e.g.*, at page 4, lines 8-9; page 7, lines 8-10; Figure 4 (426).
[8] *See* present application, *e.g.*, at page 4, lines 15-19; page 19, lines 1-8.
[9] *See id.*, *e.g.*, at Figure 1 (160); Figure 2 (202); Figure 3 (306); Figure 4 (408); Figure 6 (607).
[10] *See id.*, *e.g.*, at page 4, lines 4-6; page 7, lines 4-5; Figure 1 (100); Figure 2 (201); Figure 3 (303); Figure 4 (402); Figure 6 (604).
[11] *See id.*, *e.g.*, at Figure 4 (426).
[12] *See id.*, *e.g.*, at Figure 3 (304); Figure 4 (422); Figure 6 (605).
[13] *See id.*, *e.g.*, at page 4, lines 6-8; page 7, lines 6-7.

Claims 6-8 are dependent upon claim 5.

Independent claim 9 recites the following:

A system for communicating information in a server platform,[15] the system comprising:

a first multiserver platform[16] comprising a network interface[17] and a first switch blade;[18] and

at least a second multiserver platform[19] comprising a second switch blade[20] coupled[21] to said first switch blade[22] of said first multiserver platform.[23]

Claims 10-15 are dependent upon claim 9.

---

[14] *See* present application, *e.g.,* at page 4, lines 8-9; page 7, lines 8-10; Figure 4 (426).
[15] *See* present application, *e.g.,* at page 4, line 20.
[16] *See id., e.g.,* at Figure 1 (100); Figure 2 (201); Figure 3 (303); Figure 4 (402); Figure 6 (604).
[17] *See id., e.g.,* at Figure 1 (160).
[18] *See id., e.g.,* at page 4, lines 21; page 7, lines 15-19; page 8, lines 24-26; page 9, lines 1-2; page 10, lines 24-25; page 12, lines 2-4; page 15, lines 19-20; Figure 1 (140); Figure 2 (202); Figure 3 (306); Figure 4 (408); Figure 6 (607).
[19] *See id., e.g.,* at Figure 3 (304); Figure 4 (422); Figure 6 (605).
[20] *See id., e.g.,* at Figure 3 (307); Figure 4 (428); Figure 6 (608).
[21] *See id., e.g.,* at Figure 3 (310); Figure 4 (440).
[22] *See id., e.g.,* at Figure 1 (140); Figure 2 (202); Figure 3 (306); Figure 4 (408); Figure 6 (607).
[23] *See id., e.g.,* at page 4, lines 22-23; page 11, lines 3-5; page 12, lines 13-14; Figure 1 (100); Figure 2 (201); Figure 3 (303); Figure 4 (402); Figure 6 (604).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL (37 C.F.R. § 41.37(c)(1)(vi))

Claims 1-15 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Publication No. 2002/0188718, by McGraw et al. *See* Final Office Action at pages 4-7.

ARGUMENT
(37 C.F.R. § 41.37(c)(1)(vii))

In the Final Office Action, claims 1-15 stand rejected under 35 U.S.C. § 102(e) as being anticipated by McGraw.

I. Claims 1-15 Are Not Anticipated by McGraw

Claims 1-15 stand rejected under 35 U.S.C. § 102(e) as being anticipated by McGraw.

A. Rejection of Independent Claims 1, 5 and 9

The Appellant turns to the rejection of claim 9 under 35 U.S.C. § 102(e) as being anticipated by McGraw. The Appellant submits that McGraw does not disclose or suggest at least the limitations of "a first multiserver platform comprising **a network interface <u>and</u> a first <u>switch</u> blade**," and "at least a second multiserver platform comprising a second **<u>switch</u>** blade coupled to said first **<u>switch</u>** blade of said first multiserver platform," as set forth in Appellant's independent claim 9.

The Final Office Action alleges that McGraw's Figures 1 and 7 and Paragraphs [0138]-[0144] teach "a first multiserver platform comprising **<u>at least one</u>** of a network interface and a first switch blade."[24] However, Appellant's independent claim 9 recites "a first multiserver platform comprising **a network interface <u>and</u> a first <u>switch</u> blade**." In other words, the Final Office Action fails to make a *prima facie* case of anticipation at

[24] Final Office Action, Page 5, Line 18 – Page 6, Line 3.

least because the Final Office Action fails to address the actual limitations of Appellant's independent claim 9. The Appellant respectfully notes that Final Office Action's failure to address the Appellant's claim limitations was indicated in the Appellant's July 6, 2010 Response; however, the Advisory Action mailed July 16, 2010 failed to address the Appellant's argument.

Further, although McGraw teaches that its link card/board is a network interface card,[25] nowhere in McGraw is there any disclosure regarding a switch blade **in addition to** McGraw's link card/board (i.e., network interface card). Put another way, Appellant's independent claim 9 recites, among other things, both a network interface and a first switch blade. The Appellant notes that Appellant's network interface and first switch blade are separate and distinct elements. As such, McGraw's mere disclosure of its link card/board cannot teach both a network interface and a first switch blade.

Additionally, with regard to the Final Office Action's citation to McGraw's Figure 7 and Paragraphs [0138]-[0144] and [0159-0161], it appears the Final Office Action is alleging that McGraw's link cards/boards are **switch** blades; however, nowhere in McGraw is there any disclosure regarding McGraw's link cards/boards performing any switching functions. In fact, nowhere in McGraw do the terms "switch" and "switching" appear in McGraw. Rather, McGraw describes its link cards/boards as network interface cards.[26] One of ordinary skill in the art would readily understand that the disclosure of network interface cards is different than Appellant's claimed **switch**

[25] *See e.g.*, McGraw, Figures 2 and 5 (124), Paragraph [0128].
[26] *See e.g.*, McGraw, Figures 2 and 5 (124), Paragraph [0128].

blades. For example, as described in McGraw's Paragraph [0144], McGraw's link cards/boards merely pass messages between inter-chassis RS-485 bus and local RS-485 bus.[27] In other words, McGraw's link cards/boards are merely interfaces between inter-chassis and local buses, which are different than **switch** blades.

Also, the Appellant notes that Final Office Action-cited Figure 1 of McGraw fails to show multiserver platforms and switch blades, let alone "a first multiserver platform comprising a network interface and a first switch blade," and "at least a second multiserver platform comprising a second switch blade coupled to said first switch blade of said first multiserver platform," as set forth in Appellant's independent claim 9. Instead, McGraw's Figure 1 and its supporting disclosure merely disclose a network 30 having computing devices 32-35 having associated consoles 36-39, memory modules 45-48 and interfaces 40-43, which connect the computing devices 32-35 to a console server 50 via communication links 52-55.[28]

The Appellant next turns to the rejections of claims 1 and 5 under 35 U.S.C. § 102(e) as being anticipated by McGraw. The Appellant submits that McGraw does not disclose or suggest at least the limitations of "receiving at least one packet from at least a first **switch** blade associated with a first multiserver platform," "**determining** at least **a server blade** associated with a second multiserver platform **for receiving at least a portion of said received at least one packet**," and "**routing** said at least a portion of

[27] *See e.g.*, McGraw, Paragraph [0144].
[28] *See e.g.*, McGraw, Figure 1 and Paragraph [0025].

said at least one received packet to at least said server blade," as set forth in Appellant's independent claims 1 and 5.

Similar to the Final Office Action's allegations with regard to Appellant's independent claim 9, the Final Office Action alleges that McGraw's Figures 1 and 7 and Paragraphs [0128]-[0131] teach "receiving at least one packet from at least a first switch blade associated with a first multiserver platform," as set forth in Appellant's independent claims 1 and 5.[29] As noted above, with regard to McGraw's Figure 7 and Paragraphs [0128]-[0131], it appears the Final Office Action is alleging that McGraw's link card/board is a **switch** blade; however, nowhere in McGraw is there any disclosure regarding McGraw's link card/board performing any switching functions. In fact, nowhere in McGraw do the terms "switch" and "switching" appear in McGraw. Rather, McGraw describes its link card/board as network interface card.[30] One of ordinary skill in the art would readily understand that the disclosure of a network interface card is different than Appellant's claimed **switch** blade. For example, as described in McGraw's Paragraph [0144], McGraw's link card/board merely pass messages between inter-chassis RS-485 bus and local RS-485 bus.[31] In other words, McGraw's link card/board is merely an interface between inter-chassis and local buses, which is different than a **switch** blade.

[29] Final Office Action, Page 3, Lines 1-2.
[30] *See e.g.*, McGraw, Figures 2 and 5 (124), Paragraph [0128].
[31] *See e.g.*, McGraw, Paragraph [0144].

Also, the Appellant notes that McGraw's Figure 1 fails to show a multiserver platform and a first switch blade, let alone "receiving at least one packet from at least a first switch blade associated with a first multiserver platform," as set forth in Appellant's independent claims 1 and 5. Instead, McGraw's Figure 1 and its supporting disclosure merely disclose a network 30 having computing devices 32-35 having associated consoles 36-39, memory modules 45-48 and interfaces 40-43, which connect the computing devices 32-35 to a console server 50 via communication links 52-55.[32]

Regarding "**determining** at least **a server blade** associated with a second multiserver platform **for receiving at least a portion of said received at least one packet**," and "**routing** said at least a portion of said at least one received packet to at least said server blade," the Final Office Action alleges that McGraw's Figure 7 and Paragraphs [0138]-[0144] and [0159]-[0161] teach the Appellant's claim limitations.[33] However, nowhere in the cited section of McGraw (or elsewhere in McGraw) is there any disclosure regarding determining a server blade to route a received packet. Instead, McGraw merely teaches "[t]he Inter-chassis Link Board is responsible for forwarding the queries across the inter-chassis RS-485 bus and collects the responses from the inter-chassis RS-485 bus and conveys it on the local-485 bus to the console server."[34] McGraw's disclosure regarding merely passing messages between inter-chassis RS-485 bus and local RS-485 bus does not teach "**determining** at least **a server blade** associated with a second multiserver platform **for receiving at least a**

[32] *See e.g.*, McGraw, Figure 1 and Paragraph [0025].
[33] Final Office Action, Page 4, Lines 18-22.
[34] *See e.g.*, McGraw, Paragraph [0144], Lines 3-7.

portion <u>of said received at least one packet</u>," and "**<u>routing</u>** said at least a portion of said at least one received packet to at least said server blade," as set forth in Appellant's independent claims 1 and 5.

Clearly, McGraw merely teaches network interface cards for passing messages between inter-chassis RS-485 bus and local RS-485 bus.[35] Therefore, McGraw fails to disclose "a first multiserver platform comprising **a network interface <u>and</u> a first <u>switch</u> blade**," and "at least a second multiserver platform comprising a second **<u>switch</u>** blade coupled to said first **<u>switch</u>** blade of said first multiserver platform," as set forth in Appellant's independent claim 9; and, "receiving at least one packet from at least a first **<u>switch</u>** blade associated with a first multiserver platform," "**<u>determining</u>** at least **a server blade** associated with a second multiserver platform **for receiving at least a portion <u>of said received at least one packet</u>**," and "**<u>routing</u>** said at least a portion of said at least one received packet to at least said server blade," as set forth in Appellant's independent claims 1 and 5.

Accordingly, independent claims 1, 5 and 9 are not anticipated by McGraw and are allowable. Furthermore, the Appellant reserves the right to argue additional reasons beyond those set forth herein to support the allowability of claims 1, 5 and 9.

[35] *See e.g.*, McGraw, Figures 2, 5 and 7 (124); Paragraph [0128], Lines 9-11; Paragraph [0144], Lines 3-7.

B. Examiner's Response to Arguments

The Examiner responded to the Appellant's arguments on pages 2-3 of the Final Office Action and on page 2 of the Advisory Action.

The Response to Arguments section of the Final Office Action cites to Paragraphs [29] and [31] of the Appellant's Specification in alleging that "McGraw's link card/ board is similar in hardware (plug-in card) and function (provide connectivity between blade server and a network)."[36] The Appellant notes, however, that the Final Office Action ignores the sections of the Appellant's Specification that discuss the switching functions performed by Appellant's switch blade and central switch.[37] Nowhere in McGraw is there any disclosure that McGraw's link card/board provides any switching functions whatsoever. In fact, the terms "switch" and "switching" do not even appear in McGraw.

One of ordinary skill in the art would readily understand that just because a component is embodied as a plug-in card does not necessarily make it a **switch** blade. Similarly, one or ordinary skill in the art would readily understand that just because a component provides connectivity between a blade server and a network does not necessarily indicate that the component is a **switch** blade. Rather, one of ordinary skill in the art would readily understand that a **switch** blade provides, among other things, switching functionality. As such and as discussed above with regard to McGraw's

[36] Final Office Action, Pages 2-3.
[37] *See e.g.*, Appellant's Specification, Paragraphs [32], [40]-[41], [44]-[45], [47] and [49].

disclosure regarding its link card/board merely passing messages between inter-chassis RS-485 bus and local RS-485 bus, McGraw's link card/board is clearly not a **switch** blade.

The Advisory Action states that "Office personnel are to give claims their broadest reasonable interpretation in light of the supporting disclosure. In re Morris, 127 F.3d 1048, 1054-55, 44 USPQ2d 1023, 1027-28 (Fed. Cir. 1997)."[38] However, the Appellant notes that the "broadest construction rubric coupled with the term 'comprising' does not give the PTO an unfettered license to interpret claims to embrace anything remotely related to the claimed invention. Rather, claims should always be read in light of the specification and teaching in the underlying patent."[39] Further, "the PTO's 'broadest' interpretation must be reasonable, and must be in conformity with the invention as described in the specification."[40] Clearly, the Final Office Action's and Advisory Action's "interpretation" that a reference that fails to even mention the terms "switch" and "switching" teaches a **switch** blade is clearly unreasonable.

Accordingly, independent claims 1, 5 and 9 are not anticipated by McGraw and are allowable. Furthermore, the Appellant reserves the right to argue additional reasons beyond those set forth herein to support the allowability of claims 1, 5 and 9.

---

[38] Advisory Action, Page 2, Lines 4-5.
[39] *See In re Suitco Surface, Inc.*, 2010 U.S. App. LEXIS 7620 (Fed. Cir. April 14, 2010).
[40] *In re Ravi Vaidyanathan*, Case No. 09-1404 (C.A. Fed, May 19, 2010).

C. Rejection of Dependent Claims 2-4, 6-8 and 10-15

Claims 2-4, 6-8 and 10-15 depend on independent claims 1, 5 and 9, respectively. Therefore, the Appellant submits that claims 2-4, 6-8 and 10-15 are allowable over the reference cited in the Final Office Action at least for the reasons stated above with regard to claims 1, 5 and 9. The Appellant further submits that each of dependent claims 2-4, 6-8 and 10-15 is independently allowable.

For example, with regard to Appellant's dependent claims 2 and 6, McGraw at least fails to teach, for example, "receiving said at least one packet by at least a second switch blade associated with a third multiserver platform and a central switch." The Final Office Action cites to McGraw's Figures 1 and 7 and Paragraphs [0138]-[0144] and [0159]-[0161] as teaching the Appellant's claim limitations;[41] however, the Appellant notes that, as discussed above, nowhere in McGraw is there any disclosure regarding any switch blades or switching functions. Further, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), the Appellant notes that nowhere in McGraw is there any disclosure regarding a central switch. Specifically, as discussed in the Appellant's Specification, a central switch is not part of any multiserver platform.[42] Rather, one of the stated advantages in the Appellant's Specification of the central switch configuration is that a packet of data can be sent from a source

[41] Final Office Action, Page 3, Lines 8-17 and Page 4, Line 23 – Page 5, Line 3.
[42] *See e.g.*, Appellant's Specification, Figure 6 and Page 16, Lines 6-14.

multiserver platform to a destination multiserver platform without having to pass though intermediate multiserver platform(s). As such, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), McGraw's middle chassis's link board/card in McGraw's Figure 7 cannot be a central switch at least because it is part of the middle chassis.

As another example, with regard to dependent claims 3 and 7, McGraw at least fails to teach "if said at least one packet is received by said central switch," "communicating said at least a portion of said at least one received packet to at least a third switch blade associated with said second multiserver platform via at least one communication link that couples said central switch directly to said at least said third switch blade." The Final Office Action cites to McGraw's Figures 1 and 7 and Paragraphs [0138]-[0144] and [0159]-[0161] as teaching the Appellant's claim limitations;[43] however, the Appellant notes that, as discussed above, nowhere in McGraw is there any disclosure regarding any switch blades or switching functions. Further, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), the Appellant notes that nowhere in McGraw is there any disclosure regarding a central switch coupled to one or more of the switch blades of the multiserver platforms. Specifically, as discussed in the Appellant's Specification, a central switch is not part of any multiserver platform.[44] Rather, one of the stated advantages in the Appellant's Specification of the central switch configuration is that a

[43] Final Office Action, Page 3, Lines 8-17 and Page 5, Lines 4-9.
[44] *See e.g.*, Appellant's Specification, Figure 6 and Page 16, Lines 6-14.

packet of data can be sent from a source multiserver platform to a destination multiserver platform without having to pass though intermediate multiserver platform(s). As such, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), McGraw's middle chassis's link board/card in McGraw's Figure 7 cannot be a central switch at least because it is part of the middle chassis.

Additionally, dependent claims 3 and 7 recite a central switch in addition to three (3) multiserver platforms, each multiserver platform comprising a switch blade. In contrast, McGraw's Figure 7 merely illustrates three (3) chassis, each comprising a link board/card, which is different than the Appellant's claim limitations. Also, McGraw explicitly teaches that its Figure 7 is implemented in a daisy-chain configuration,[45] which indicates that even if McGraw did disclose an additional chassis, McGraw's daisy chain configuration would be incapable of teaching three multiserver platforms communicating via a central switch.[46]

With regard to Appellant's dependent claim 13, McGraw at least fails to teach, for example, "at least one central switch coupled to at least said first switch blade of said first multiserver platform and said second switch blade of said second multiserver platform." The Final Office Action cites to McGraw's Figures 1 and 7 and Paragraphs [0138]-[0144] and [0159]-[0161] as teaching the Appellant's claim limitations;[47] however, the Appellant notes that, as discussed above, nowhere in McGraw is there any

[45] McGraw, Figure 7; Paragraph [0159], Lines 6-7.
[46] *See e.g.*, Appellant's Specification, Page 16, Lines 7-14.
[47] Final Office Action, Page 3, Lines 8-17 and Page 6, Lines 13-16.

disclosure regarding any switch blades or switching functions. Further, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), the Appellant notes that nowhere in McGraw is there any disclosure regarding a central switch coupled to at least said first switch blade of said first multiserver platform and said second switch blade of said second multiserver platform. Specifically, as discussed in the Appellant's Specification, a central switch is not part of any multiserver platform.[48] Rather, one of the stated advantages in the Appellant's Specification of the central switch configuration is that a packet of data can be sent from a source multiserver platform to a destination multiserver platform without having to pass though intermediate multiserver platform(s). As such, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), McGraw's middle chassis's link board/card in McGraw's Figure 7 cannot be a central switch at least because it is part of the middle chassis.

With regard to Appellant's dependent claim 14, McGraw at least fails to teach, for example, "at least a third switch blade of a third multiserver platform coupled to said at least one central switch." The Final Office Action cites to McGraw's Figures 1 and 7 and Paragraphs [0138]-[0144] and [0159]-[0161] as teaching the Appellant's claim limitations;[49] however, the Appellant notes that, as discussed above, nowhere in McGraw is there any disclosure regarding any switch blades or switching functions. Further, even if McGraw's link cards/boards could be considered switch blades (which

---

[48] *See e.g.*, Appellant's Specification, Figure 6 and Page 16, Lines 6-14.
[49] Final Office Action, Page 3, Lines 8-17 and Page 6, Line 17 – Page 7, Line 2.

they clearly are not), the Appellant notes that nowhere in McGraw is there any disclosure regarding a central switch coupled to at least said first switch blade of said first multiserver platform, said second switch blade of said second multiserver platform, and a third switch blade of a third multiserver platform. Rather, McGraw's Figure 7 merely illustrates three (3) chassis, each comprising a link board/card, which is different than the Appellant's claim limitations. Also, McGraw explicitly teaches that its Figure 7 is implemented in a daisy-chain configuration,[50] which indicates that even if McGraw did disclose an additional chassis, McGraw's daisy chain configuration would be incapable of teaching three multiserver platforms communicating via a central switch.[51]

Further, as discussed in the Appellant's Specification, a central switch is not part of any multiserver platform.[52] Rather, one of the stated advantages in the Appellant's Specification of the central switch configuration is that a packet of data can be sent from a source multiserver platform to a destination multiserver platform without having to pass though intermediate multiserver platform(s). As such, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), McGraw's middle chassis's link board/card in McGraw's Figure 7 cannot be a central switch at least because it is part of the middle chassis.

With regard to Appellant's dependent claim 15, McGraw at least fails to teach, for example, "wherein said first multiserver platform, said second multiserver platform and

[50] McGraw, Figure 7; Paragraph [0159], Lines 6-7.
[51] *See e.g.*, Appellant's Specification, Page 16, Lines 7-14.
[52] *See e.g.*, Appellant's Specification, Figure 6 and Page 16, Lines 6-14.

said third multiserver platform communicate via said central switch." The Final Office Action cites to McGraw's Figures 1 and 7 and Paragraphs [0138]-[0144] and [0159]-[0161] as teaching the Appellant's claim limitations;[53] however, the Appellant notes that, as discussed above, nowhere in McGraw is there any disclosure regarding any switch blades or switching functions. Further, even if McGraw's link cards/boards could be considered switch blades (which they clearly are not), the Appellant notes that nowhere in McGraw is there any disclosure regarding said first multiserver platform, said second multiserver platform and said third multiserver platform communicating via said central switch. Rather, McGraw's Figure 7 merely illustrates three (3) chassis, each comprising a link board/card, which is different than the Appellant's claim limitations. Also, McGraw explicitly teaches that its Figure 7 is implemented in a daisy-chain configuration,[54] which indicates that even if McGraw did disclose an additional chassis, McGraw's daisy chain configuration would be incapable of teaching three multiserver platforms communicating via a central switch.[55]

Further, as discussed in the Appellant's Specification, a central switch is not part of any multiserver platform.[56] Rather, one of the stated advantages in the Appellant's Specification of the central switch configuration is that a packet of data can be sent from a source multiserver platform to a destination multiserver platform without having to pass though intermediate multiserver platform(s). As such, even if McGraw's link

[53] Final Office Action, Page 3, Lines 8-17 and Page 7, Lines 3-6.
[54] McGraw, Figure 7; Paragraph [0159], Lines 6-7.
[55] *See e.g.*, Appellant's Specification, Page 16, Lines 7-14.
[56] *See e.g.*, Appellant's Specification, Figure 6 and Page 16, Lines 6-14.

cards/boards could be considered switch blades (which they clearly are not), McGraw's middle chassis's link board/card in McGraw's Figure 7 cannot be a central switch at least because it is part of the middle chassis.

Accordingly, the Appellant submits that claims 2-4, 6-8 and 10-15 are allowable over the reference cited in the Final Office Action at least for the above reasons. The Appellant also reserves the right to argue additional reasons beyond those set forth above to support the allowability of claims 2-4, 6-8 and 10-15.

CONCLUSION

For at least the foregoing reasons, the Appellant submits that claims 1-15 are in condition for allowance. Reversal of the Examiner's rejection and issuance of a patent on the application are therefore requested.

The Commissioner is hereby authorized to charge $540 (to cover the Brief on Appeal Fee) and any additional fees or credit any overpayment to the deposit account of McAndrews, Held & Malloy, Ltd., Account No. 13-0017.

Respectfully submitted,

Date: 27-OCT-2010

By: /Philip Henry Sheridan/
Philip Henry Sheridan
Reg. No. 59,918
Attorney for Appellant

McANDREWS, HELD & MALLOY, LTD.
500 West Madison Street, 34th Floor
Chicago, Illinois 60661
(T) 312 775 8000
(F) 312 775 8100

(PHS)

CLAIMS APPENDIX
(37 C.F.R. § 41.37(c)(1)(viii))

1. A method for communication of information in a server platform, the method comprising:

receiving at least one packet from at least a first switch blade associated with a first multiserver platform;

determining at least a server blade associated with a second multiserver platform for receiving at least a portion of said received at least one packet; and

routing said at least a portion of said at least one received packet to at least said server blade.

2. The method according to claim 1, wherein said receiving further comprises receiving said at least one packet by at least a second switch blade associated with a third multiserver platform and a central switch.

3. The method according to claim 2, further comprising if said at least one packet is received by said central switch, communicating said at least a portion of said at least one received packet to at least a third switch blade associated with said second multiserver platform via at least one communication link that couples said central switch directly to said at least said third switch blade.

4. The method according to claim 1, further comprising processing said routed at least a portion of said at least one received packet by said at least said server blade.

5. A machine-readable storage having stored thereon, a computer program having at least one code section for communicating information in a server platform, the at least one code section being executable by a machine for causing the machine to perform steps comprising:

receiving at least one packet from at least a first switch blade associated with a first multiserver platform;

determining at least a server blade associated with a second multiserver platform for receiving at least a portion of said received at least one packet; and

routing said at least a portion of said at least one received packet to at least said server blade.

6. The machine-readable storage according to claim 5, further comprising code for receiving said at least one packet by at least a second switch blade associated with a third multiserver platform and a central switch.

7. The machine-readable storage according to claim 6, further comprising code for communicating said at least a portion of said at least one received packet to at least a third switch blade associated with said second multiserver platform via at least one communication link that couples said central switch directly to said at least said third switch blade, if said at least one packet is received by said central switch.

8. The machine-readable storage according to claim 5, further comprising code for processing said routed at least a portion of said at least one received packet by said at least said server blade.

9. A system for communicating information in a server platform, the system comprising:

a first multiserver platform comprising a network interface and a first switch blade; and

at least a second multiserver platform comprising a second switch blade coupled to said first switch blade of said first multiserver platform.

10. The system according to claim 9, further comprising at least a third multiserver platform comprising a third switch blade coupled to at least said second switch blade of said second multiserver platform and said first switch blade of said first multiserver platform.

11. The system according to claim 10, wherein said first multiserver platform, said second multiserver platform and said third multiserver are coupled in a daisy-chain configuration.

12. The system according to claim 10, wherein said first multiserver platform, and said third multiserver platform communicate via said second multiserver platform.

13. The system according to claim 9, further comprising at least one central switch coupled to at least said first switch blade of said first multiserver platform and said second switch blade of said second multiserver platform.

14. A system according to claim 13, further comprising at least a third switch blade of a third multiserver platform coupled to said at least one central switch.

15. The system according to claim 14, wherein said first multiserver platform, said second multiserver platform and said third multiserver platform communicate via said central switch.

EVIDENCE APPENDIX
(37 C.F.R. § 41.37(c)(1)(ix))

(1) United States Publication No. 2002/0188718 ("McGraw"), entered into record by the Appellant in the Information Disclosure Statement mailed April 27, 2005.

RELATED PROCEEDINGS APPENDIX
(37 C.F.R. § 41.37(c)(1)(x))

The Appellant is unaware of any related appeals or interferences.

EXHIBIT 1

US 20020188718A1

(19) United States
(12) Patent Application Publication (10) Pub. No.: US 2002/0188718 A1
McGraw et al. (43) Pub. Date: Dec. 12, 2002

(54) CONSOLE INFORMATION STORAGE SYSTEM AND METHOD

(75) Inventors: **Montgomery C. McGraw**, Spring, TX (US); **Ramkrishna V. Prakash**, Houston, TX (US); **David P. Sharp**, Houston, TX (US); **Lazaro D. Perez**, Houston, TX (US)

Correspondence Address:
Baker Botts L.L.P.
Suite 600
2001 Ross Ave.
Dallas, TX 75201-2980 (US)

(73) Assignee: **RLX Technologies, Inc.**

(21) Appl. No.: **10/039,051**

(22) Filed: **Dec. 31, 2001**

Related U.S. Application Data

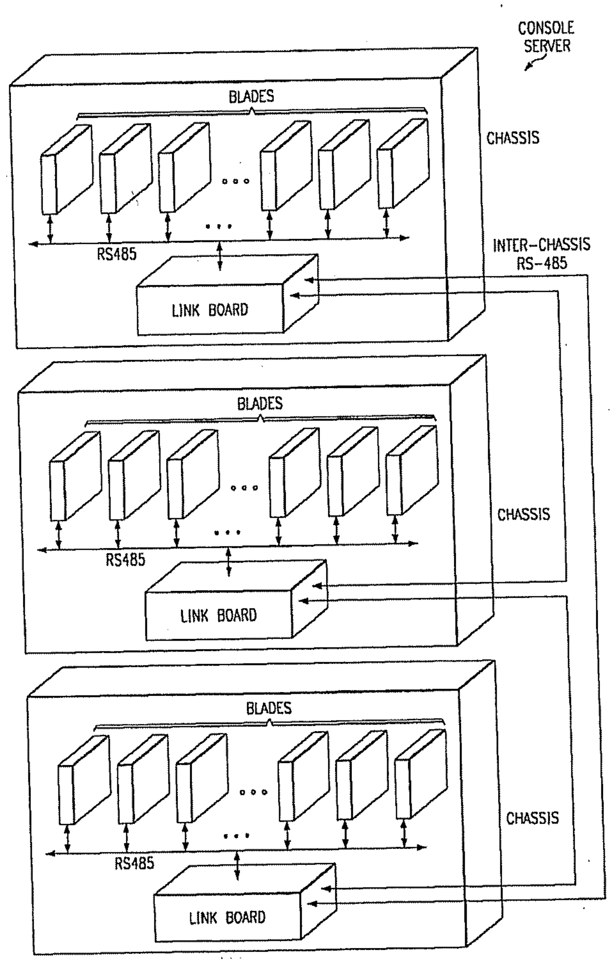
(60) Provisional application No. 60/288,614, filed on May 4, 2001.

Publication Classification

(51) Int. Cl.[7] **G06F 15/173**
(52) U.S. Cl. **709/224**

(57) **ABSTRACT**

A system and method for storing console information includes a first computing device having a first console and a first console interface operable to transmit first console information associated with the first console. A second computing device is coupled for communication with the first computing device. The second computing device may include a memory module operable to receive the first console information. In a particular embodiment, the memory module may be operable to store the first console information.



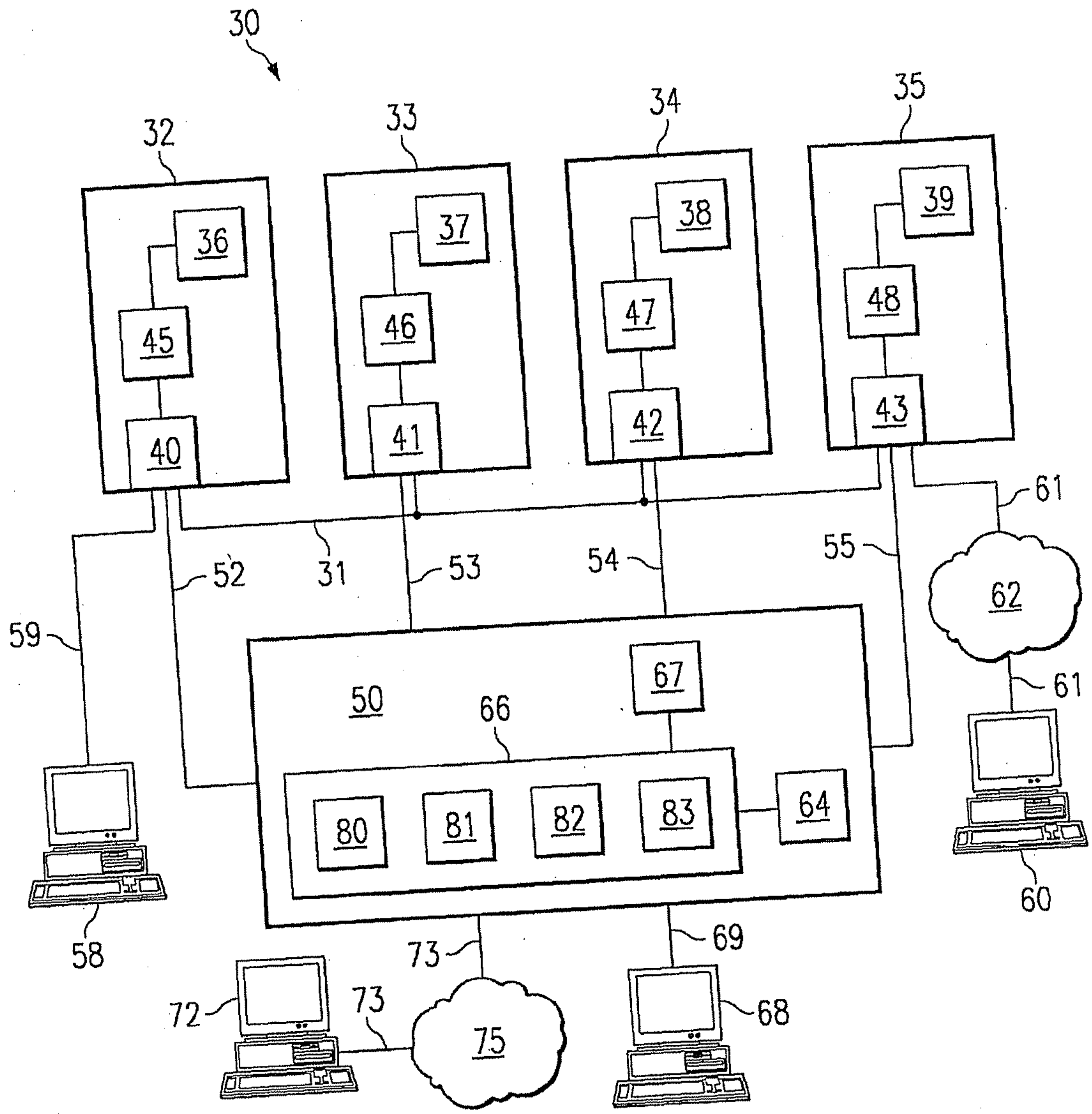

30
32
33
34
35
36
37
38
39
45
46
47
48
40
41
42
43
61
31
52
53
54
55
62
59
50
66
67
61
80
81
82
83
64
58
60
73
69
72
73
75
68

FIG. 1

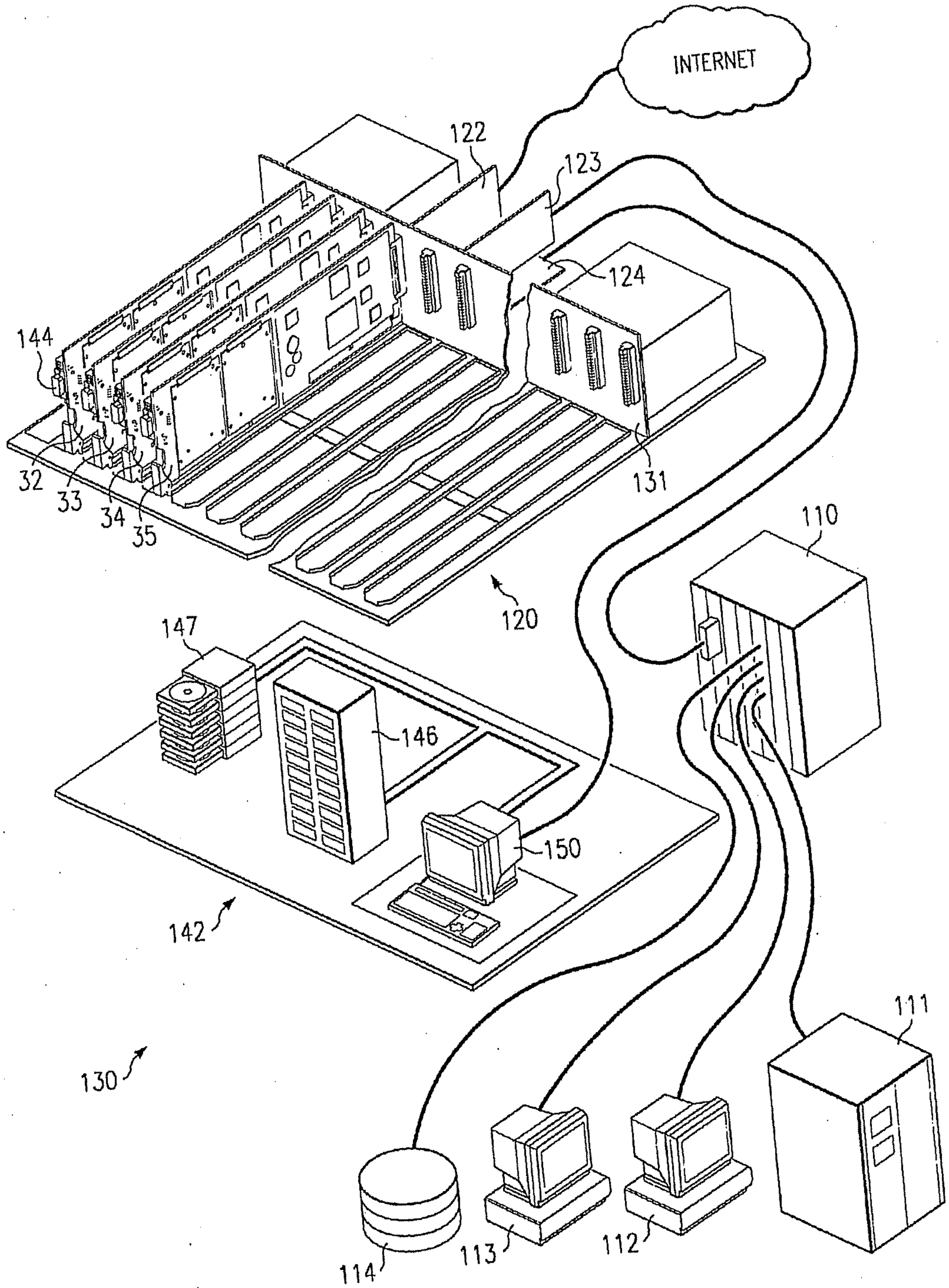

INTERNET
122
123
124
144
131
32
33
34
35
120
110
147
146
150
142
130
111
114
113
112

FIG. 2

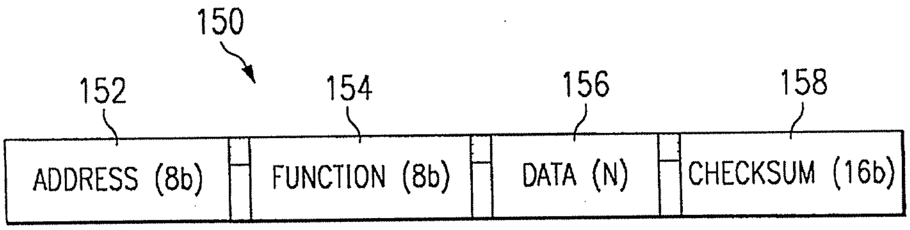


FIG. 3

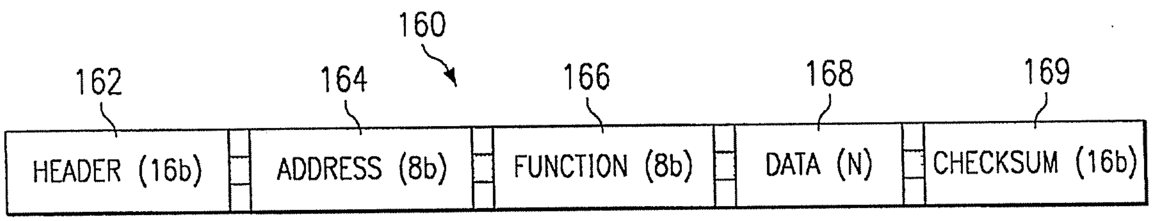


FIG. 4

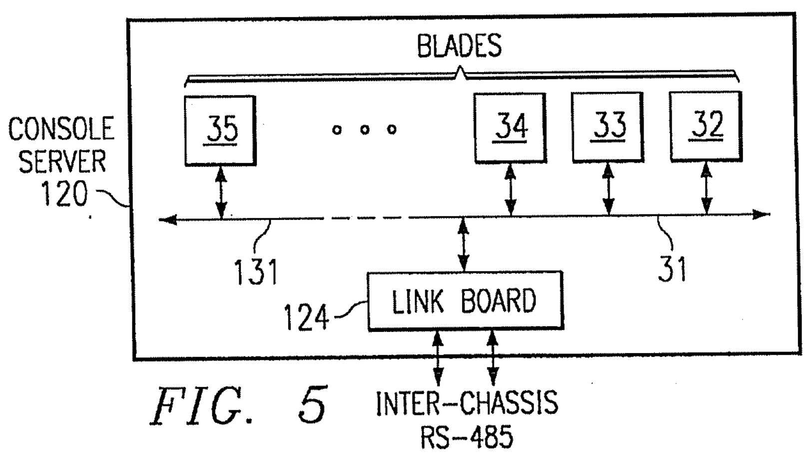
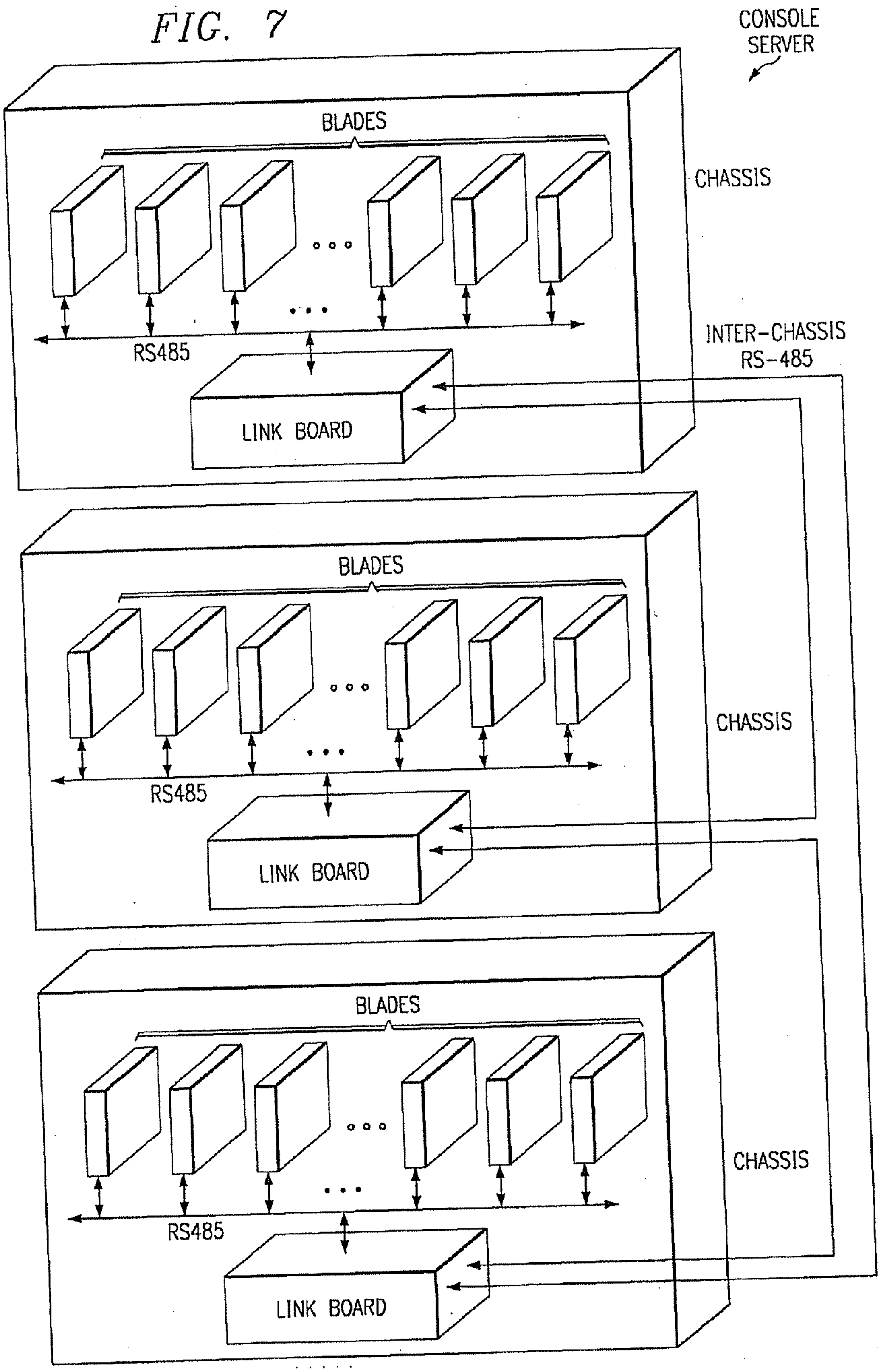


FIG. 5

	MANAGEMENT BOARD	SLOT 1	SLOT 2	BACKUP CS
CASE 1	CPU – PRESENT (CS)	PRESENT	PRESENT	SLOT 1
CASE 2	CPU – PRESENT (CS)	ABSENT	PRESENT	SLOT 2
CASE 3	ABSENT	PRESENT (CS)	PRESENT	SLOT 2
CASE 4	~~PIC~~ CPU – PRESENT	PRESENT (CS)	PRESENT	SLOT 2

FIG. 6

FIG. 7


CONSOLE SERVER
BLADES
CHASSIS
RS485
LINK BOARD
INTER-CHASSIS RS-485
BLADES
CHASSIS
RS485
LINK BOARD
BLADES
CHASSIS
RS485
LINK BOARD

7	6	5	4	3	2	0
CMD	SCOPE	MULTI-BYTE	RSVD	RSVD	TASKS	200

FIG. 8

FIELDS	VALUE	FUNCTION
CMD	0	THIS IMPLIES IT IS A COMMAND MESSAGE
SCOPE	0 OR 1	0: IT IS A CHASSIS LOCAL MESSAGE 1: IT IS AN INTER CHASSIS MESSAGE NOTE: WHEN THE SCOPE IS DEFINED AS LOCAL ONLY THE SLOT ID FIELD IS PRESENTED IN THE <Slot-Identifier>. ON THE OTHER HAND WHEN THE SCOPE IS DEFINED AS INTER CHASSIS BOTH THE SLOT ID AND CHASSIS ID FIELDS ARE PRESENTED IN THE <Slot-Identifier>
MULTI-BYTE	0 OR 1	0: NO DATA BYTES FOLLOWING 1: DATA BYTES FOLLOWING
RSVD	UNASSIGNED	
RSVD	UNASSIGNED	
TASKS	00H-07H	00: RECEIVE COMMAND 220 01: TRANSMIT COMMAND 222 02: IDENTIFY COMMAND 224 03: STATUS COMMAND 226 04: Re_sync 228 05: SET 230 06: GET 232 07: IDENTIFY CHASSIS COMMAND 234

FIG. 9

7	6	5	4	3	2	1	0
ACK	SCOPE	MULTI-BYTE	EXTENDED	RSVD	ERROR	WATERMARK	DATA
RSVD	RSVD	RSVD	RSVD	FNC NOT SUPPORTED	NAK	HEALTH	OVERRUN

FIG. 10A

FIG. 10B

FIELDS	VALUE	FUNCTION
ACK	1	THIS IMPLIES IT IS AN ACKNOWLEDGE MESSAGE
SCOPE*	0 OR 1	0: IT IS A CHASSIS LOCAL MESSAGE 1: IT IS AN INTER CHASSIS MESSAGE NOTE: WHEN THE SCOPE IS DEFINED AS LOCAL ONLY THE SLOT ID FIELD IN THE PRESENTED IN THE <Slot-Identifier>. ON THE OTHER HAND WHEN THE SCOPE IS DEFINED AS INTER CHASSIS BOTH THE SLOT ID AND CHASSIS ID FIELDS ARE PRESENTED IN THE <Slot-Identifier> *SLAVE BLADES NEED NOT SET THIS FIELD. IT IS INTENDED TO BE ADDED BY INTER-CHASSIS COMMUNICATION BOARD ALONG WITH THE CHASSIS ID AND SLOT ID
MULTI-BYTE	0 OR 1	0: NO DATA BYTES FOLLOWING 1: DATA BYTES FOLLOWING
EXTEND ACK	0 OR 1	0: IMPLIES ACK FIELD IS ONLY 1 BYTE 1: IMPLIES ACK FIELD IS 2 BYTES
RSVD	UNASSIGNED	RESERVED FOR FUTURE USE
ERROR	0 OR 1	0: NO ERROR IN PREVIOUS COMMAND 1: ERROR IN PREVIOUS COMMAND
WATERMARK	0 OR 1	0: DATA IS LESS THAN 3/4 SIZE OF AVAILABLE BUFFER 1: DATA IS LESS THAN 3/4 SIZE OF AVAILABLE BUFFER
DATA	0 OR 1	0: NO DATA AVAILABLE FOR TRANSFER 1: DATA AVAILABLE FOR TRANSFER

FIG. 10C

FIELDS	VALUE	FUNCTION
RSVD	UNASSIGNED	RESERVED FOR FUTURE USE
RSVD	UNASSIGNED	RESERVED FOR FUTURE USE
RSVD	UNASSIGNED	RESERVED FOR FUTURE USE
RSVD	UNASSIGNED	RESERVED FOR FUTURE USE
FNC NOT SUPPORTED	0 OR 1	0: THE FUNCTION REQUESTED WAS O.K. 1: THE FUNCTION REQUESTED IS NOT SUPPORTED ON THE BLADE
NACK	0 OR 1	0: THE BLADE CAN RESPOND TO MASTER'S REQUEST 1: THE BLADE NOTIFIES THE MASTER TO REQUEST AT A LATER TIME AS THE BLADE IS BUSY
HEALTH	0 OR 1	0: CPU NOT FUNCTIONAL (SLEEP) 1: CPU FUNCTIONAL
OVERRUN	0 OR 1	0: DATA IN BUFFER HAS NOT BEEN OVERRUN 1: DATA IN BUFFER HAS BEEN OVERRUN

CONSOLE INFORMATION STORAGE SYSTEM AND METHOD

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. provisional application Ser. No. 60/288,614 filed May 4, 2001.

[0002] This application is filed concurrently with the following commonly owned patent application entitled Console Information Server System and Method (Attorney's Docket 067856.0235).

TECHNICAL FIELD OF THE INVENTION

[0003] The present invention relates generally to server chassis communication systems and, more particularly, to a console information storage system and method.

BACKGROUND OF THE INVENTION

[0004] Computing devices typically include a console that is used to control the computing device manually, correct errors, manually revise the contents of storage, and provide communications in other ways between an operator and the central processing unit and/or operating system. Console interfaces provoke an interface between the console of a computing device and the operator, or an external device.

[0005] A user interface (e.g., graphical user interface) may be coupled with the console interface to allow a local user to access the console of the computing device. The user interface may be used to provide a visible representation of information, whether in words, numbers, and/or drawings, on a user interface coupled with the computing device. User interfaces may include graphical user interfaces, monitors, keyboards, etc.

[0006] Console information generated by the console includes data, communications and/or signals communicated between the console of the computing device and an operator. Such information typically includes health, administrative, configuration and/or programming information, tools, commands, data and other information. Console information may also include data, signals, commands and other communications from a terminal unit to a console. For example, during startup of a personal computer, console information is displayed at a monitor coupled with the computing device. The console information includes health and configuration information regarding the particular computing device, its operating system, hardware and/or software components.

SUMMARY OF THE INVENTION

[0007] The present invention provides a system and method for storing console information associated with the console of a computing device. In accordance with a particular embodiment of the present invention, console information from one or more computing devices is collected and stored at a memory module, and made available for future presentation to an operator.

[0008] In one embodiment, a computing device includes a console and a console interface operable to transmit console information associated with the console. A memory module operable to receive the console information is also included. In a particular embodiment, the memory module is further operable to store the console information for retrieval by an operator of the computing device.

[0009] In accordance with another embodiment of the present invention, a system includes a first computing device having a first console and a first console interface operable to transmit first console information associated with the first console. A second computing device is coupled for communication with the first computing device. The second computing device includes a memory module operable to receive the first console information. The memory module may be further operable to store the first console information.

[0010] In still another embodiment of the present invention, a third computing device is coupled for communication with the second computing device. The third computing device includes a second console and a second console interface operable to transmit second console information associated with the second console. The memory module may be further operable to receive and store the second console information.

[0011] Technical advantages of the present invention include a system and method for storing console information generated by the console of a computing device. By collecting, storing, manipulating, and/or presenting the console information to an operator, performance of the computing device(s) and associated console(s) may be reviewed and analyzed in the future.

[0012] Another technical advantage of the present invention includes a system and method for collecting console information from a plurality of computing devices, at a central location. In this manner, a single computing device may be used to collect, store, and analyze console information from a plurality of computing devices. The storage of this information allows later presentation to an operator.

[0013] Other technical advantages will be readily apparent to one skilled in the art from the following Figures, descriptions, and claims. Moreover, while specific advantages have been enumerated above, various embodiments may include all, some or none of the enumerated advantages.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] For a more complete understanding of the present invention and its advantages, reference is now made to the following descriptions, taken in conjunction with the accompanying drawings, in which:

[0015] **FIG. 1** illustrates a communication network including a server and a plurality of computing devices, incorporating various aspects of the present invention;

[0016] **FIG. 2** illustrates a communication network including web server processing cards and network interface cards of a server chassis coupled for communication with various network components, in accordance with a particular embodiment of the present invention;

[0017] **FIG. 3** illustrates the structure of a communication frame which may be used in accordance with a particular embodiment of the present invention;

[0018] **FIG. 4** illustrates the structure of a communication frame which may be used in conjunction with a particular embodiment of the present invention;

[0019] FIG. 5 is a block diagram illustrating communication between a plurality of computing devices and a link board, in accordance with a particular embodiment of the present invention;

[0020] FIG. 6 illustrates a selection chart for mapping responsibility of a plurality of computing devices, in accordance with a particular embodiment of the present invention;

[0021] FIG. 7 is a block diagram illustrating inter and intra communication between a plurality of server chassis, in accordance with a particular embodiment of the present invention;

[0022] FIG. 8 illustrates bit field command messages, in accordance with a particular embodiment of the present invention;

[0023] FIG. 9 is a graphical representation illustrating the definition of the bit fields of FIG. 8, in accordance with a particular embodiment of the present invention; and

[0024] FIG. 10 is a graphical representation of bit fields of Acknowledge messages and their potential values, in accordance with a particular embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0025] FIG. 1 is a schematic drawing illustrating a communication network 30 in accordance with a particular embodiment of the present invention. Network 30 includes a plurality of computing devices 32-35, each having an associated console 36-39, respectively, and console interface, 40-43, respectively. Memory modules 45-48 are coupled with consoles 36-39, respectively and are operable to store console information regarding computing devices 32-35, respectively. Each console interface 36-39 is coupled with a console server 50, using a plurality of communication links 45-48. In a particular embodiment of the present invention, console information regarding computing devices 32-35 is stored at memory modules 45-48, respectively, and/or communicated to console server 50. Accordingly, historical and/or real-time console information regarding computing devices 32-35 may be accessed by users of network 30. Furthermore, console server 50 provides access to computing devices 32-35 for communicating console information to and from computing devices 32-35 for monitoring, debugging, troubleshooting, maintenance, configuration and/or updates to terminal units 32-35.

[0026] Throughout this application, the term "console" includes the section of a computing device that is used to control the computing device manually, correct errors, manually revise the contents of storage, and provide communications in other ways between the operator and the central processing unit and/or operating system. Console interfaces 40-43 provide an interface between the console of each computing device and an operator and/or external device. A user interface may be coupled with the console interface to allow a local user to access the console of the computing device. For example, a console display 58 may be coupled with console interface 40. Console display 58 may include a visible representation of information, whether in words, numbers, and/or drawings, on a console screen coupled with computing device 32. In various embodiments of the present invention, console display 58 may include a user interface, graphical user interface, monitor, keyboard, mouse, personal computer and/or other computing devices.

[0027] Throughout this application, the term "console information" includes any data, communication, and/or signals communicated between the console of the computing device and an operator and/or user. Console information typically includes health, administrative, configuration and/or programming information, tools, commands, data and other information. Console information also includes data, signals, commands and other communications from a terminal unit to a console. During startup of a standard personal computer (PC), console information is displayed at a monitor coupled with the computing device. The console information includes health and configuration information regarding the particular computing device, it's operating system, hardware and/or software components. However, this information is not stored for later retrieval. Therefore, if a monitor is not coupled with the computing device, the console information cannot be viewed by a user. Furthermore, the user must view the console information in real time, as it is communicated from the computing device.

[0028] The manner in which console information is communicated and/or displayed to a user depends, at least in part, upon the particular software, hardware, and/or configuration of the computing device. For example, particular versions of the Microsoft Windows operating system are configured to display console information to a user of an IBM compatible PC via a video graphics array (VGA) interface. The Linux operating system, on the other hand, typically displays console information to a user in a serial manner. Regardless of the configuration of hardware and/or software associated with the computing device, the teachings of the present invention provide a method for storing, at least temporarily, this console information for later retrieval by a user.

[0029] In the illustrated embodiment, the console information may also be communicated to a server or another computing device coupled with the computing device of interest. For example, the console information regarding computing device 32 may be communicated to console server 50 and/or one or more of computing devices 33-35, which may be coupled with computing device 32. The server or attached computing device may also be configured to store the console information regarding one or more computing devices.

[0030] Memory module 45 includes hardware, software, and/or logic operable to read, record, buffer, store and/or communicate data and information between and among components internal and external to computing device 32. Since each computing device 32-35 includes similar components and function similarly, the operation and functionality of computing device 32 will be described in detail. However, it shall be recognized that all aspects and functionality of components of computing device 32 pertains to each computing device 33-35. For example, each memory module 46-48 is configured and functions similarly to memory module 45, with regard to their respective computing devices 33-35, respectively.

[0031] Console information associated with computing device 32 is communicated to console interface 36. Therefore, if console display 58 is coupled with console interface 40, a user can view the console information in "real-time" at

the user interface. In accordance with a particular embodiment of the present invention, memory module **45** receives all console information generated by computing device **32** and stores the console information. Memory module **45** may comprise a buffer. Accordingly, memory module **45** would include a finite capacity. Therefore, data (e.g. console information) is stored until the buffer is full. When memory module **45** reaches its capacity, it begins writing over the oldest data currently in the buffer.

[0032] Console information regarding computing device **32**, which is stored within memory module **45**, may be accessed for further processing, by a user of network **30**. For example, a user may couple a terminal unit **58** with computing device **32** and retrieve the console information stored at buffer **40**. Console information stored within memory module **45** may be referred to as "historical console information."

[0033] Throughout this application, the term "real-time" console information includes console information which is read and/or displayed as it is received from console **36**. On the other hand, "historical console information" includes stored console information which is read, and stored by memory module **45**, and/or any console information that is not communicated in real-time. Unless otherwise specified, the term console information shall mean real-time console information, historical console information, and/or any other information stored in memory module **45** (e.g., alerts), throughout this application. The term "memory module" may include all types of memory and storage media operable to store data, at least temporarily, for retrieval by a user and/or another computing device or server. In particular embodiments, memory module **45** may include random access memory (RAM), read only memory (ROM), dual in-line memory modules (DIMMs), registers, buffers, integrated circuits, volatile memory, micro-programmable devices, disk subsystems, and/or non-volatile memory. The term "module" includes software, hardware, and/or encoded logic operable to read, record, store, buffer, and/or communicate data and information between and among components of network **30**.

[0034] Memory module **45** includes historical console information regarding computing device **32**. In other words, memory module **45** includes console information collected (read) and stored over a period of time. Therefore, a user of terminal unit **58** may view console information communicated from console **36** before the occurrence of a specific event. For example, if computing device **32** experiences trouble, or crashes during operation, the user of terminal unit **58** may view console information communicated before, during and/or after the event. Similarly, the user of terminal unit **58** can review console information communicated by console **36** in the past, in order to determine the reaction of computing device **32** to any specific event or particular operating conditions and/or characteristics. This type of historical console information was not previously available to a user of a computing device. Instead, real-time console information was available to the computer operator if a user interface was coupled with computing device **32** and the console information was viewed by the operator in real-time, as it was communicated from console **36** to the user interface.

[0035] As illustrated in **FIG. 1** with regard to computing device **35**, a user may access real-time and/or historical console information regarding console **39**, from a remote location. A terminal unit **60** is coupled with computing device **35** using communication link **61**. Communication link **61** extends through communication network **62**. Accordingly, a user may access console interface **43** from a remote location and view real time console information as it is received from console **39**. The user may establish a two-way communication session in order to communicate with console **39**. Alternatively, the user of terminal unit **60** may communicate with memory module **48** in order to retrieve historical console information stored within memory module **48**. In alternative embodiments, terminal unit **60** may also be coupled for communication with one or more of computing devices **32-34** in order to monitor, review, and administer each computing device **32-35**, remotely.

[0036] In accordance with another embodiment of the present invention, real-time console information generated by computing device **32** is communicated to console server **50** using communication link **52**. Data and information received at console server **50**, including real-time console information received from computing devices **32-35**, may be viewed at a user interface **64** of console server **50**. Console server **50** may be local to computing devices **32-35** (e.g. located on the same premises) or console server **50** may be located at a remote location, and/or coupled with computing devices **32-35** through a communication network.

[0037] Console server **50** includes a memory module **66** which is operable to store the console information received from computing device **32**. Data and information stored in memory module **66**, including historical console information received from computing devices **32-35** may be viewed at user interface **64**. Therefore, a user of console server **50** may view historical console information regarding console **36** of computing device **32**, by accessing memory module **66**. Console server **50** also includes a console **67**. Console server **50** may be configured to collect, read, buffer, store, process, communicate, and/or control its own console **67** and/or console information associated with console **67**.

[0038] In another embodiment, for example if console server **50** does not include user interface **64**, a terminal unit **68** may be used to view the console information received from computing device **32**. Terminal unit **68** is coupled with console server **50** using communication link **69**. Terminal unit **68** may be used to view real time console information received by console server **50** from computing device **32**, in real time as it is received at console server **50**. Alternatively, terminal unit **68** may be used to review historical console information regarding computing device **32**, which is stored within memory **66**.

[0039] A terminal unit **72** is coupled with console server **50** using a communication link **73**. Communication link **73** extends through a communication network **75**. Therefore, a user of terminal unit **72** may access console server **50** from a remote location, through network **75**. The user of terminal unit **72** has access to real-time console information as it is communicated from console **36** to console server **50** and to terminal unit **72**. The user of terminal unit **72** also has access to historical console information stored in any of memory modules **45-48** and/or memory module **66**.

[0040] A plurality of terminal units, computing devices, user interfaces and servers, are disclosed throughout this

specification. Any component included with one of these devices may also be included with any other, or all such devices. In alternative embodiments, terminal units, computing devices, user interfaces, monitors, and/or servers may include telephones, computers, personal computers, laptops, notebook computers, personal digital assistants, keyboards, monitors, memory modules, consoles, console interfaces, and any components associated therewith capable of data communication, and/or data processing internally, locally, and/or over a network. Communication links and communication networks disclosed herein may include any computer and/or communication network including, without limitation, the public switched telephone network (PSTN), the Internet, intranets, local area networks (LANs), wide area networks (WANs), or metropolitan area networks, for wireless or wireline communication incorporating twisted pair, cable, optical fiber, or other suitable wireline links, and/or radio frequency, microwave, infrared and/or other suitable wireless links.

[0041] In accordance with a particular embodiment of the present invention, memory module **66** may be configured to "poll" consoles **36-39** and/or memory modules **45-48** periodically, in order to collect and/or store real-time and/or historical console information associated with computing devices **32-35**. In a particular embodiment, console server **50** communicates with one or more of computing devices **32-35**, at predetermined time intervals, to collect real-time and/or historical console information. The console information collected by console server **50** may be stored at memory module **66** for retrieval by a network component coupled with console server **50**. In other embodiments, console information may be communicated from computing devices **32-35** to console server **50** by interrupt driven/on-demand requests from either the computing devices or the console server. In other words, the console server may be configured to request console information from the computing devices in response to a particular event, circumstance, alert or situation. Similarly, the computing devices may be configured to transmit console information to the console server in response to a particular event, circumstance, alert or situation.

[0042] Memory module **66** includes a plurality of buffer modules **80-83**, which communicate with computing devices **32-35**, respectively. Therefore, console information collected by console server **50** regarding computing devices **32-35** may be partitioned, for convenient access to console information regarding a particular computing device, by users of console server **50**, terminal unit **68** and/or terminal unit **72**.

[0043] In accordance with another embodiment, computing devices **32**, consoles **36-39** and/or memory modules **45-48** may be configured to transmit real-time and/or historical console information to console server **50** continuously, or at predetermined time intervals. There are many methods of communication between and among the various components of network **30**. It will be recognized by those of ordinary skill in the art that any communication between components which result in real-time and/or historical console information being communicated to console server **50** and available for retrieval by a user of network **30** is included with aspects of the present invention.

[0044] Terminal units **68, 72** and/or console server **50** may be used to establish two way communication with computing devices **32-25**, their respective consoles **36-39**, and /or memory modules **45-49**. Therefore, console **50**, terminal unit **68**, and/or terminal unit **72** may be used to collect and transmit information to any particular component of consoles **36-39**. Accordingly, such users can perform operation, administration, troubleshooting, maintenance, debugging, and/or updates of consoles **36-39** from a remote location.

[0045] For example, a user of console server **50** may display, in a single communication session, console information regarding a single computing device. Alternatively, the user may display console information regarding all computing devices, simultaneously, in a single communication session. Furthermore, the user may select a group including two or more particular computing devices of computing devices **32-35** to communicate with in a single communication session.

[0046] This feature allows the user to review the console information associated with the group of computing devices to determine how each reacts/and or reacted to a particular situation. For example, if one or more servers crash, the user can review and/or compare the console information from each computing device that crashed, and/or perform maintenance, debugging, and/or repair on such terminal units simultaneously.

[0047] Console interface **40** is configured to broadcast communication sessions with terminal unit **58**, and to console server **50**, in real-time. Similarly, communication sessions between console server **50** and computing device **32** are communicated to terminal unit **58**, in real time. Therefore, if a local user couples terminal unit **58** with console interface **40**, and begins a communication session with console **36** and/or memory module **45**, the communication session may be viewed, in real time, at user interface **64**, terminal unit **68**, and/or terminal unit **72**. This allows two users to communicate with computing device **32** simultaneously, and each can view exactly what the other is doing and/or seeing at their respective user interface during their respective communication sessions. Accordingly, two users in remote locations from one another may cooperate to simultaneously communicate with and/or debug a particular computing device and/or group of computing devices.

[0048] Each computing device **32-35** of the illustrated embodiment is coupled with a communication bus **31**. In the illustrated embodiment, bus **31** comprises an RS-485, two wire bus. In alternative embodiments, bus **31** may include RS-232, Ethernet, USB, and/or any communication link. Console information and other data, signals, and/or other information may be communicated between and among computing devices **32-35** using communication bus **31**. Accordingly, one of computing devices **32-35** may be configured to function as the console server. If a computing device is selected to perform the functionality of console server **50**, that computing device may include some or all of the components disclosed herein with regard to console server **50**. Console information regarding one or more of computing devices **32-35** may be collected, and/or stored at a particular of computing devices **32-35**.

[0049] **FIG. 2** illustrates a communication network **130**, in accordance with a particular embodiment of the present invention. Network **130** includes a server chassis **120**, with portions broken away for clarity, having the plurality of computing devices **32-35** coupled with a midplane **131**.

Midplane **131** includes communication bus **31** which couples server processing cards **32-25**. In a particular embodiment, computing devices **32-35** include server processing cards.

[0050] A plurality of network interface cards **122-124** are coupled with midplane **131** and bus **31**, and provide server processing cards **32-35** with access to a plurality of communication network components. For example, network interface card **122** is coupled with the Internet **140**. Network interface card **123** couples server chassis **120** with another network of components **110-114**. Network interface card **124** couples server processing cards **32-35**, midplane **131** and network interface cards **122-123** with a management network **142**.

[0051] Management network **142** includes console server **50** coupled with a plurality of network attached storage (NAS) devices **146-147**. Management network **142** may be used for remote monitoring, configuration, debugging, maintenance, and/or management of server processing cards **32-35** and/or network interface cards **122-124**.

[0052] In a particular embodiment, network interface cards **122-124** may include a "daughter" card(s) which comprise identical components and functionality to computing devices **32-35** and/or console server **50**. Accordingly, all of the components and functionality of console server **50** may be attached directly to network interface card **124**. Alternatively, all of the components of console server **50** may be attached directly to any one of computing devices **32-35**. Therefore, a network operator may select the console server from among any computing device, network interface card, local terminal unit, and/or remote terminal unit.

[0053] As previously discussed, computing devices **32-35** may comprise server processing cards providing access to various communication networks, including the Internet. In a particular embodiment, a network administrator may distribute memory and processing power associated with server processing cards **32-35** to various customers. Such customers may use server processing cards **32-35** for storage of data, computing and processing of data, and/or web site hosting. The network operator may assign each customer to a different server processing card **32-35**, or multiple customers may share one or more server processing cards **32-35**.

[0054] In one embodiment, each computing device **32-35** may include at least two microprocessors, wherein one of the microprocessors is dedicated to perform the console memory function. Accordingly, the main CPU of each computing device will not be burdened with tasks of collecting, manipulating, storing and/or communicating console information. This approach provides transparency to the computing device's main CPU BIOS and OS, as opposed to performing console memory functions using the main CPU BIOS and OS. Furthermore, this type of architecture may be combined with the architecture of **FIG. 7** in such a way that all console information is collected, manipulated, stored and/or communicated to other computing devices and/or servers without using the main CPU BIOS or OS.

[0055] In a particular embodiment, all of the components associated with console server **50** may be attached directly to one of server processing cards **32-35**, allowing that particular card to assume console server responsibilities with regard to the other server processing cards, and network interface cards **122-124**. If the network administrator selects server processing card **32**, for example, to function as the console server for a given session, server processing card **32** will communicate with, and collect console information associated with server processing cards **33-35**. This allows the network administrator to consolidate all console information with regard to server processing cards **32-35** upon a single server processing card. A local user may access server processing card **32** by coupling a terminal unit with a console interface **144**. A user may also access console information regarding server processing cards **32-35** by coupling a terminal unit with network interface card **124**. Similarly, a user may access console information regarding server processing cards **32-35** by remotely coupling a terminal unit with network interface card **124** and communicating the server processing card **32**.

[0056] Two users may communicate with server processing card **32** regarding console information associated with server processing cards **32-35**, simultaneously. In other words, if a user is coupled with console interface **144** locally, and communicating console information with server processing card **32**, a user of a second console server, for example, console server **50**, may view this communication session in its entirety, and participate. If the user at console server **50** communicates information with server processing card **32**, the user coupled with console interface **144** can view this communication session, and participate. This feature provides simultaneous debugging of a server processing card or cards, wherein both users are local to chassis **120**, one user is local and one user is remote, and/or both users are remote to server chassis **120**.

[0057] All of the components and functionality associated with console server **50** may also be attached directly to network interface card **124**. In this embodiment, network interface card **124** may be referred to as a management network interface card. Console information regarding server processing cards **32-35** may be communicated with management network interface card **124**, and accessed by a user by coupling a terminal unit directly with management network interface card **124**, locally, or remotely. All of the components, features, and functionality of console server **50** described herein may be included with one or more server processing cards **32-35** and/or one or more network interface cards **122-124**.

[0058] In a particular embodiment, console server **50** communicates with server processing cards **32-35** and network interface cards **122-124**, in order to determine which components are present. Console server **50** sends a message to each component asking that component to identify itself. Console server **50** then maintains a log of all components present during a session. Console server **50** periodically polls each of computing devices **32-35** and/or management network interface card **124**, regarding console information. Console server **50** sends a message to the device commanding that device to forward console information stored at that device. Console server **50** collects the console information, and makes it available to a user of console server **50**.

[0059] In another embodiment, all console information regarding server processing cards **32-35** and/or management interface card **124**, are collected at a single device, for example, server processing card **32** or management network

interface card **124**. In this embodiment, console server **50** communicates with server processing card **32** or management network interface card **124** periodically to collect all console information regarding server processing cards **32-35** and/or management network interface card **124**.

[**0060**] In a particular embodiment, a backup console server may be selected from among server processing cards **32-35**, management network interface card **124**, and/or console server **50**. A backup console server may be configured to detect communication from the primary console server. If the backup console server does not detect communication from the primary console server for a predetermined amount of time, the backup server will execute an error message. The backup server will then either request permission from a network operator to assume console server responsibilities, and/or immediately begin performing console server responsibilities with regard to all components present. Alternatively, the backup console server may be configured to perform redundant console server responsibilities, in order to prevent loss of data due to failure of the primary console server. In this embodiment, the backup console server will collect console information regarding server processing cards **32-35** and/or management interface card **124**, and include duplicate information to the primary console server.

[**0061**] Server processing cards which are not functioning as a primary or backup console server may be referred to as slave server processing cards, for example, server processing cards **33-35**. Slave server processing cards are operable to buffer their own console information and provide data to the console server at predetermined intervals, and/or upon request of the console server.

[**0062**] In another embodiment, multiple server chassis may be provided, each having multiple server processing cards associated therewith. Multiple server chassis may be coupled using a communication bus, such as communication bus **31**. The coupling between multiple server chassis may be accomplished by coupling the communication bus with network interface cards in each attached server chassis. In a particular embodiment, the communication bus may comprise an RS-485 bus. Accordingly, console server **50** may be operable to collect console information from a plurality of server processing cards distributed amongst a plurality of server chassis. In this manner, all console information regarding any number of server processing cards associated with any number of server chassis may be consolidated at console server **50**.

[**0063**] Console server **50** may also be used to communicate with and/or configure, debug, and/or operate any of the server processing cards. This allows a user of console server **50** to establish a communication session with a plurality of server processing cards distributed amongst a plurality of server chassis during a single session. For example, the user could command console server **50** to display all console information regarding one, all or a specified group of server processing cards, in a single session. The user could also execute commands to all, or a subset of all server processing cards simultaneously. For example, in a particular embodiment, console server **50** may be used to reset all server processing cards, or a subset of server processing cards with a single command. The reset is operable to cause a reboot of each server processing card. The reboot may be from an operating system resident upon the particular server processing cards. Alternatively, console server **50** may be used to issue a command to all, or a subset of all server processing cards to boot from an attached element associated with a local area network (LAN). Console server **50** may also issue a command for a computing device "wake" (similar to wake on lan command). Similarly, console server **50** may issue a command to put the computing device to "sleep".

[**0064**] Console server **50** may also monitor CPU and system health associated with components of computing devices **32-35**. Accordingly, console server **50** provides for "out-of-band" management of computing devices **32-35**.

[**0065**] Console server **50** may include security software which allows access to a particular server processing card or group of server processing cards to a single user only. The security may include a password input by the user at console server **50**. Accordingly, console server **50** may be configured to allow different users access to different server processing cards. This feature prevents a user of one server processing card from gaining access to information upon another server processing card.

[**0066**] Console server **50** is also configured to allow a particular user to name their server processing cards, using alphanumeric letters. Communication with server processing cards is typically established with identification numbers. However, a user of console server **50** may name a particular server processing card or group of server processing cards. Therefore, a user who has access to three server processing cards **33-35** may name one of the servers "web server," one of the servers "credit card database," and one of the servers "storage database." This is a user-friendly feature which allows a user to easily establish which server processing card they need access to perform a particular function or functions. An alphanumeric prompt may also be displayed while displaying console information from a particular computing device, which identifies the particular computing device. The prompt may be color coded to identify the state (communicating information, not communicating console information) and indicate which computing device is currently transferring information.

[**0067**] A user who has access to server processing cards **32-35** may request console server **50** to display all console information with regard to server processing cards **32-35** simultaneously upon a user interface of console **50**. In this manner, the user can compare and contrast console information to determine how each server processing card is affected by a particular set of circumstances or operating conditions. Such conditions may include software updates, and/or intense processing loads experienced by the computing device(s). The user may also execute commands to server processing cards **32-35**, simultaneously, over console server **50**. Alternatively, the user may command console server **50** to display console information or allow communication with a single console associated with any of server processing cards **32-35**. The user can execute commands at consoler server **50** which allow the user to "toggle" between various server processing cards. In other words, during a communication session with console **37** of server processing card **33**, the user may interrupt that session without losing information, and toggle to a communication session with console **38** associated with server processing card **34**. The user could also access console **39** associated with server

processing card **35** without losing any information from the communication session with server processing cards **33** and **34**.

[**0068**] Console server **50** provides a unified console for a user to access any computing device **32-35** that console server **50** serves. Accordingly, console server **50** provides an integrated interface that enables a user to use or access the interface from a variety of devices. Console server **50** provides flexibility to streamline access to multiple consoles. At least two modes of operation are available for console server **50**: (i) command line mode; (ii) shell mode.

[**0069**] In the command line mode, a user who logs into console server **50** is able to run command line queries. This mode provides a quick and efficient manner to gather data from various computing devices **32-35**. The privilege associated with access of information is based on log in privileges that the user is assigned as a result of logging into console server **50**.

[**0070**] If a user issues a particular command, for example, "RLXCONSOLE," the user enters the shell mode of console server **50**. Options available to a user from the command line mode include the following:

[**0071**] 1. rlxconsole–list(l)

[**0072**] rlxconsole–list(l)—chassis(cn)<chassis_number>

[**0073**] rlxconsole–list(l)–chassis(cn)<chassis_number>–slot(sn)<slot_name>

[**0074**] rlxconsole–list(l)–group(g)<file_name>

[**0075**] When used with just the –list option, all the computing devices **32-35** console server **50** can allow the user to access are listed. When used in conjunction with the –chassis option, only the computing devices **32-35** in the specified chassis **120** are listed. When used with the additional option of –slot, only the particular computing devices **32-35** identified is listed. If the user does not have privileges to access either the chassis **120**, the specified computing devices **32-35**, an error notification message is posted.

[**0076**] The –group accompanied with a filename option used in conjunction with the –list option list all the computing devices **32-35** identified in the filename. The –group option allows the user to create a group of computing devices **32-35** on which the user would like certain operations performed. The file that contains the list of the computing devices **32-35** is a simple text file with each computing devices **32-35** identified by its chassis number and slot number separated by colons. Any text preceded by a "#" sign until a return character is deemed as a comment. If the user does not have privilege to access a particular computing devices **32-35** listed in the group file, an error notification is posted.

[**0077**] 2. rlxconsole–status(s)

[**0078**] rlxconsole–status(s)–chassis(cn <chassis_number>

[**0079**] rlxconsole–status(s)–chassis(cn)<chassis_number>–slot(sn)<slot_name>

[**0080**] rlxconsole–status(s)–group(g)<file_name>

[**0081**] This option lists the status of all the computing devices **32-35** that the console server **50** can allow the user to access. The status information indicates which of the possible computing devices **32-35** are really capable of being accessed. When used in conjunction with the –chassis option, only the status of computing devices **32-35** in the specified chassis are listed. When used with the additional option of –slot, the status of only the particular computing device **32-35** identified is listed. If the user does not have privileges to access either the chassis **50** or the specified computing device **32-35**, an error notification message is posted. The status option can also be used in conjunction with the –group option.

[**0082**] 3. rlxconsole–backup(b)

[**0083**] rlxconsole–backup(b)–chassis(cn)<chassis_number>–slot(sn)<slot_name>

[**0084**] This option when used by itself identifies the currently designated backup console server. When used in conjunction with the additional options –chassis<chassis_number>–slot<slot_name> to identify a particular computing device **32-35**, it results in the specified computing device **32-35** being designated as the new backup console server and reassignment of the previous backup console server to be just a computing device **32-35**. Only the "root" level user has privileges to execute this command.

[**0085**] 4. rlxconsole

[**0086**] rlxconsole–chassis(cn)<chassis_number>–slot(sn)<slot_number>

[**0087**] rlxconsole–chassis(cn)<chassis_number>

[**0088**] rlxconsole–group(g)<filename>

[**0089**] The rlxconsole command when issued by itself without any options, results in invoking of the rlxconsole shell, which will be discussed later. When used with additional options –chassis and –slot to identify a particular computing device **32-35**, a console session is established with the computing device **32-35** within the rlxconsole shell. If just the –chassis option is used, then console sessions to each of the computing devices **32-35** accessible to the user is made within the rlxconsole shell. However, only the computing device **32-35** with the lowest slot ID is displayed. If the user wished to access the console server **50** of any of the other computing devices **32-35** within the chassis to which the console server **50** made a connection, the user would have to use rlxconsole shell command "rlxtoggle." The shell commands are discussed in the next section. If the user used rlxconsole with the –group options, computing devices **32-35** will establish console sessions to each of the computing devices **32-35** mentioned in <filename> that is accessible to the user. Here again, the connection is made within the rlxconsole shell and only the computing devices **32-35** with lowest Chassis:Slot ID combination is displayed. Here again, using the rlxconsole shell command, the user can access other console session.

[**0090**] 5. rlxconsole–user(u)<username>

[**0091**] rlxconsole–user(u)<username>–passwd(p)<password>

[**0092**] rlxconsole–superuser(su)<password>

[0093] This option allows the user to change his access profile to that of the user specified by the username. If the specified username has privileges that are lower than the one the user used to login, a password is not required. However, if the specified username has privileges that are higher than the one the user used to login, then a password is required. When used with the option –superuser, it is assumed that user wants "root" level privileges. To assume superuser level privileges, the user must already be running in root level privilege mode on the system before evoking these privileges within rlxconsole. This option can be used in conjunction with any of the command line options discussed above.

[0094] The rlxconsole shell allows a user to engage in prolonged console sessions with computing devices **32-35** in a concurrent manner. A user can enter the rlxconsole in one of two ways described above. The first method is by executing the rlxconsole as a command line operation. The second is by evoking a command line of the rlxconsole with the option of listing one or more computing devices **32-35**. Once inside the rlxconsole shell, only shell commands are valid. These shell commands that a user can execute within the rlxconsole shell are described below:

[0095] 1. rlxconnect–chassis(cn)<chassis_number>–slot(sn)<slot_number>

[0096] rlxconnect–chassis(cn)<chassis_number>

[0097] rlxconnect–group(gn)<filename>

[0098] This command is similar to the rlxconsole command line command, but used within the rlxconsole shell. rlxconsole command does not work within the shell. The options allowed for rlxconnect are similar to the rlxconsole line command used with the same options (see item 4 cases second through fourth). Here again, the options –chassis<chassis_number>–slot<slot_number> refers to a particular computing device **32-35** in the chassis. If the user supplied just the option –chassis<chassis_number>, all the computing devices **32-35** in the chassis that the user has access to are also connected. Similarly, using the –group<filename> option, all the computing devices **32-35** listed in the file are connected. Note that only the computing devices **32-35** that the user has permission to access are connected. Also, if any error occurs, an error notification is provided.

[0099] 2. rlxtoggle

[0100] rlxtogglet–chassis(cn)<chassis_number>–slot(sn)<slot_number>

[0101] rlxtoggle–chassis(cn)<chassis_number>

[0102] rlxtoggle–slot(sn)<slot_number>

[0103] This command allows the user to toggle between the current console connections to a pre-established console connection with a computing device **32-35**. When rlxtoggle command is issued, the computing device **32-35** with the next higher chassis ID:slot ID combination, connection, is consoled into. A particular computing device **32-35** can be specified using the –chassis (cn)<chassis_number>–slot(sn)<slot_number> option. If just the –chassis (cn)<chassis_number> is issued, then it assumed that the user would like to access the console of the computing device **32-35** with the current slot ID, but with the specified chassis ID. Likewise, if just the –slot(sn)<slot_number> option is provided, then it is assumed that the user would like to console into the computing devices **32-35** with the same chassis ID, but with the specified slot ID. rlxtoggle can enable the user to console into established connection. If an user tries to toggle into computing device **32-35** who has not been connected to, then an error notification is generated.

[0104] 3. rlxexit

[0105] rlxexit–all

[0106] rlxexit–chassis(cn)<chassis_number>–slot(sn)<slot_number>

[0107] rlxexit–chassis(cn)<chassis_number>

[0108] rlxexit–slot(sn)<slot_number>

[0109] This shell command allows a user to close a console connection made to a computing device **32-35**. When rlxexit is specified without any option, then the current console connection that the user is viewing is closed. If no console connection is in progress, then this causes the user to exit from rlxconsole shell. When this command is used with the –all option, all the console connections currently established are closed, and the user exits from the console shell. If a user would like to close the connection to a particular computing device **32-35**, then the user can specify this using the –chassis cn)<chassis_number>–slot(sn)<slot_number> options. If the user uses either the –chassis(cn)<chassis_number> or the –slot(sn)<slot_number> option with the rlxexit shall command, then connection to all the computing devices **32-35** with specified chassis ID or slot ID are terminated respectively.

[0110] 4. rlxbackup

[0111] rlxbackup–chassis(cn)<chassis_number>–slot(sn)<slot_name>

[0112] This command, when used without any option lists the current backup console server. When used with the –chassis(cn)<chassis_number>–slot(sn)<slot_name> causes the identified computing devices **32-35** to be designated as the new backup console server. Note that this command will work only if the user has "root" privileges.

[0113] 5. rlxlist

[0114] rlxlist–chassis(cn)<chassis_number>–slot(sn)<slot_name>

[0115] rlxlist–chassis(cn)<chassis_number>

[0116] rlxlist–slot(sn)<slot_name>

[0117] This command allows the user to list all the computing devices **32-35** that the user has access to. The additional options allow the user to specify a particular computing device **32-35** in a chassis with a similar slot ID respectively. This command operates in a similar manner as the rlxconsole–list command line version.

[0118] 6. rlxstatus

[0119] rlxstatus–chassis(cn)<chassis_number>–slot(sn)<slot_name>

[0120] rlxstatus–chassis(cn)<chassis_number>

[0121] rlxstatus–slot(sn)<slot_name>

[0122] This command allows the user to query the status of all the computing devices **32-35** that the user has access to. The additional options allow the user to specify a particular computing device **32-35** or computing devices **32-35** in a chassis or with the similar slot ID respectively. This command operates in a similar manner as the rlxconsole–status command line version.

[0123] The communication of console and other information between computing devices **32-35** and/or console server **50** of the present invention, employs a console server protocol which is built on top of the open standard ModBus protocol, which defines a multi-dash drop protocol RS232, RS422, or RS485 over a variety of media. Such media may include, without limitation, fiber, radio, cellular, etc. In a particular embodiment of the present invention, the console server protocol is used over the RS485 physical layer.

[0124] FIGS. **3-9** and the description below illustrate particular embodiments which incorporate some aspects of the present invention.

[0125] The framing of the ModBus protocol is described in more detail, with regard to **FIG. 3**. **FIG. 3** illustrates the basic structure of a ModBus frame **150**. The ModBus packet format comprises an address header **152** followed by a function **154**, which in turn is followed by data **156**. A two-byte CRC algorithm is used to error-check this packet, and this is appending to each of the packets. The ModBus protocol comprises an eight-bit address field within address header **152**, followed by an eight-bit function field within function **154**. The data field within data **156** that follows the function field is of variable length. The ModBus frame ends with a sixteen-bit CRC **158** that is calculated over the entire packet. When this packet is transmitted over the RS485 bus, silent periods before and after the transfer are used as the delimiters for transfer.

[0126] The framing of the console server protocol is discussed in more detail with regard to **FIG. 4**. Modifications have been made to the ModBus protocol to facilitate additional flexibility and ease the processing burden on computing devices **32-35** that are not involved in a particular communication session. A delimiter field has been added to minimize overhead associated with tracking of communication periods. The start delimiter is FFFF and silent periods function as the end delimiter. Accordingly, there are no FFFF patterns in other packets. The total number of data bytes transferred in a packet of the illustrated embodiment will not exceed 64K-1 bytes. Adapting this delimiter enables the computing devices that are not involved in communication to passively monitor for a start delimiter pattern to resynchronize communication.

[0127] A typical frame of data **160** that is transferred using the console server protocol follows the general format graphically illustrated in **FIG. 4**. Header field **162** comprises the start delimiter FFFF. Address field **164** is a slot-identifier field that identifies the computer device **32-35** that the console server **50** is communicating with. The function field **166** denotes either a message from a console server **50** to a computing device **32-35**, or a response from computing device **32-35** to console server **50**. Data field **168** is optional and is dependent on the function field **166** type. The interpretation of data field **168** is dependent on the function field **166**. The last field is a sixteen-bit CRC **169**, which is calculated over the entire packet. The physical protocol over which the data will be transferred in the illustrated embodiment, comprises the RS485 protocol. In this embodiment, the RS485 protocol requires the transfers of one byte of data at a time. Each byte transfer using the RS485 protocol begins with a start bit followed by a byte of data, a paradity bit and a stop.

[0128] An overview of the console server behavior is discussed in more detail with regard to **FIG. 5**. **FIG. 5** includes console server **50**, which is capable of communicating with computing devices **32-35** and the link computing device in its chassis to gather data as well as to determine the states of each computing device. The link card in a chassis is the computing device which communicates with link cards and/or computing devices of other chassis to collect information from computing devices within other chassis. In a particular embodiment, the link card comprises network interface card **124** of **FIG. 2**. The link card entity is a bridge between multiple chassis and is involved in proxy of commands on behalf of console server **50**. This enables console server **50** to provide integrated console services across multiple chassis.

[0129] A slave blade refers to a computing device that communicates with console server **50** when it is communicated with by console server **50**. The slave blade may be any computing device **32-35** that is not acting as console server **50**. If designated as a backup console server, the slave blade is capable of monitoring the activity of console server **50**, and capable of taking over in situations when console server **50** is not functional. For example, if the slave blade does not receive communication from console server **50** for a predetermined period, the slave blade may be configured to take over the function of console server **50**. **FIG. 5** illustrates a schematic view of these components and associated interconnection paths.

[0130] Console server **50** is responsible for collecting console information and data from the slave blades. The slave blades in turn respond to commands issued by console server **50** to transfer data to console server **50**. Console server **50** is also responsible for sending console data to the appropriate slave blades.

[0131] The typical sequence of messages of the protocol that console server **50** uses to communicate with the slave blades may follow the following sequence: (i) console server **50** sends a message; (ii) slave blade receives the message; (iii) slave blade sends a response; and (iv) console server receives a response.

[0132] In a particular embodiment of the present invention, the console server may support the following features. The console server may keep a history buffer of the console messages from all the computing devices **32-35** that it collects data from. The console server may communicate console data to and from all the monitor computing devices **32-35**. When a session is initiated to a computing device **32-35**, all the buffer data stored by console server **50** for that computing device will be presented. On repeated querying of a computer device **32-35**, only data not presented previously is displayed.

[0133] The operation of the console server may include at least two phases, the configuration phase and the operation phase. In the configuration phase, console server **50** determines the number of active computing devices **32-35** in its

chassis, and determines the number of active chassis in its neighborhood. The neighborhood of a particular chassis includes all other chassis which include computing devices under the monitoring or control of the console server. In one embodiment, this includes some or all of the computing devices within a given chassis. In the operation phase, the console server collects data from each of the computing devices **32-35**, which the console server detected during the configuration phase. The console server also transmits console data to the appropriate computing devices **32-35**. Periodically, the console server collects data from chassis within the console server's neighborhood over an inter-chassis RS485 bus.

[0134] The slave blade is a slave to the console server in its operation; hence, its default mode is to listen to traffic on the local RS485 bus. If the slave blade detects, or hears a command with the slot identifier that matches the chassis and slot number of the slave blade, then a response to the query of console server **50**. After responding to the query, it goes back to its default listen mode.

[0135] During the configuration phase, in order to detect computing devices **32-25** in a chassis, the console server sends the "Identify <Slot-Identifier>" command on the local RS-485 bus and listens for a response. The Slot-Identifier assumes two values based on whether the message is intra-chassis or inter-chassis. For intra-chassis communication, the Slot-Identifier field assumes the Slot Number (8b) of the computing device that the console server is communicating with. For inter-chassis communication, the Slot-Identifier field assumes the Slot Number (8b) of the Link board. The Slot number is the value of the slot wherein the computing device is plugged. The Link board (when incorporated) assumes the unique slot number F7.

[0136] The Slave Blade at the slot indicated by the Slot-Identifier on receiving the Identify message replies with the "Acknowledge <Data>" command. The Data field associated with this response comprises of the following information of the blade:

<Number of Bytes>	
<Chassis Number (8b)>	
<Slot Number (8b)>	
<Not participating (8b)>	
<PIC code version (16b)	Format: Ax(4b)x(4b).x(4b)>
<BIOS code version (16b)[2]	Format: Bx(4b)x(4b).x(4b)>
<Linux driver version (16b)	Format: Cx(4b)x(4b).x(4b)>
<Windows driver version (16b)	Format: Dx(4b)x(4b).x(4b)>
<Console Server version (16b)	Format: 1x(4b)x(4b).x(4b)>

[0137] After supplying this data, the Slave Blade goes back to its listen mode. The console server on receiving this answer makes an entry in the "Configuration Table" and continues to send the Identify command with the next Slot-Identifier. If the console server does not hear from the Slave Blade in a pre-determined time interval, it resends the "Identify <Slot-Identifier>" message on the RS-485 bus. If the console server does not hear from a Slave Blade even on its third request, it concludes that there is no Slave Blade in the slot represented by the Slot-Identifier and moves on to query the next slot. It then also updates the Configuration Table indicating board-not present in the slot.

[0138] In accordance with a particular embodiment of the present invention, computing devices (e.g. Slave blades) in neighboring chassis are detected using an embedded microprocessor based, inter-chassis communication board. On detecting an Inter-chassis Link Board on its chassis, the console server sends the Identify_Interchassis command on the local RS-485 bus. After sending this command, the console server waits for a response. The Link Board is the only card that acts on this command. On sensing the Identify_Interchassis command on the local RS-485 bus, the Link Board forwards the request to the inter-chassis RS-485 bus along the out_port and appends a <Data> field to the command. The Data in the forwarded message comprises a <number-of-chassis> field and <chassis ID> field which is the chassis ID. The subsequent Inter-chassis Link board that receives this command in turn forwards it further on but prior to that it increments the <number-of-chassis> field and appends its chassis ID to <chassis ID> field. As this messages cycles through the chassis, it finally reaches the Inter-chassis Link board on the chassis with console server. The Link Board in this Chassis then forwards the aggregated response to the console server over the local RS-485 bus.

[0139] If the console server does not receive an Identify_Interchassis response in pre-determined time interval, it resends the Identify_Interchassis command again. If it does not receive a reply even on its third attempt, it concludes that no chassis are present in its neighborhood and updates the Configuration table to indicate this. Thus, the Configuration Phase results in either the creation or the updating of the Configuration Table, which logs the boards present in its chassis and the presence of an additional neighboring chassis.

[0140] During the operation phase, communication between computing devices (e.g. server blades) within a chassis is accomplished as follows. In order to detect the states of the slave blade(s), the console server sends a "Status<Slot-Identifier>" command on the RS-485 bus and listens for "Acknowledge<Slot-Identifier>" response from a Slave Blade. The Slot-Identifier field in the Acknowledge response is slot number of the responding Slave Blade. The Acknowledge response comprises Status fields that indicates some or all of the following: Slave Blade has data; buffer overrun; buffer data has passed ¾ capacity; error was detected in the request; CPU health; multiple Byte transfer; NAK which implies the Console to retry later; function not supported; and/or extended Acknowledge.

[0141] There are two formats for the Acknowledge messages that result as a response to the Status message. The short format comprises of a one-byte Acknowledge response. The long format comprises of a two-byte Acknowledge response. Whenever the Slave blade or a console server detects an error in the message issued to them, they respond back with the Error bit set in the "Acknowledge" response. The setting or clearings of the bit fields of the Acknowledge response are discussed later in more detail.

[0142] Whenever the console server determines a Server Blade has console data and wants to get it from a Slave Blade, it sends a "Transit<Slot-Identifier>" command on the RS-485 bus and listens for a response. The Slave Blade, whose slot number matches the Slot-Identifier, on receiving this Transmit command starts to reply. The Slave Blade

replies with "Acknowledge<Slot-Identifier><Data>" command followed by data. The first two bytes of the data field indicate the number of data bytes that the Slave Blade intends on transmitting, followed by the actual data. If the Slave Blade has no data to transmit, it sets the first two bytes to zeros, indicating it intends to send no data. The Slot-Identifier indicates the slot number of the Slave Blade. The console server on successfully receiving all the bytes sent by the Slave blade sends an "Acknowledge" message with the Error bit cleared. The console server then proceeds to request data from the next Slave Blade as indicated by the Configuration Table. If, on the other hand, the console server did not receive the bytes successfully, it sends the "Acknowledge" message with the Error bit set. The Slave Blade, on receiving this message, resends the entire data. If the console server does not receive the data correctly from the Slave Blade in three tries, it logs an error.

[0143] If the console server has Console data that it needs to send to the Server Blade, it sends a "Receive <Slot-Identifier><Data." command on the RS-485 bus and listens for an acknowledgement. The Slave Blade on successfully receiving all the bytes sent by the console server sends the "Acknowledge" message with Error bit cleared. If, on the other hand, the Slave Blade did not receive the bytes successfully, it sends the "Acknowledge" message with the Error bit set. The console server on receiving this message resends the entire data.

[0144] The console server and the Serve Blades may use the same command sequence as described above for inter-chassis communication. The Inter-chassis Link Board is responsible for forwarding the queries across the inter-chassis RS-485 bus and collects the responses from the inter-chassis RS-485 bus and conveys it on the local RS-485 bus to the console server.

[0145] The Slave Blade performs the functions described below:

> **[0146]** Detects and identifies itself (Slot and Chassis ID); determines if it is designated as a backup console server; and/or responds to command issued to it by the console server. In the case that the Slave Blade has also been designated as a backup console server, it is responsible for the detection of the dysfunction of the console server and is capable of taking over the console server function.

[0147] The steps that the Slave Blade would go through are as follows:

> **[0148]** (i) On power up, the slave blade determines its identity in the chassis along with details regarding the version of software and firmware running on it. The identity is defined as the Chassis and the Slot-Number that it is in. If it determines that its Slot number is either 1 or 2 and that it's "Master bit" has not been set, it concludes that it has the potential of becoming the backup console server; (ii) if a Slave Blade is not a backup console server, then it waits for commands from the console server and responds with appropriate replies; and (iii) if the Slave Blade is also a backup console server, then in addition to replying to commands from the console server, it monitors for traffic on the local RS-485 bus. If it notices there is no activity on the Local RS-485 bus, it concludes that console server is non-operational and it performs the recovery procedure that it has been programmed to execute.

[0149] Two recovery procedures for a backup console server include manual recovery and automated recovery. For manual recovery, the backup console server on detection of a non-functional primary console server, notifies the NOC.

[0150] In the automated recovery mode, the Slave Blade, which is deemed as the backup console server, automatically takes over the functionality of the console server and activates console server software applications on its blade. Thus, the backup console server assumes the full functionality of the defunct console server.

[0151] Assignment of the console server may be done manually or automatically. In the manual mode, the operator is responsible for manually assigning which of the Slave Blades will be responsible for being the backup console server.

[0152] In accordance with a particular embodiment of the present invention, Server Blades in Slot **1** and Slot **2**, and the Inter-chassis communication Board, or link board, are capable of assuming the backup console functionality. In the automatic assignment mode, one order of preference is as listed in **FIG. 6**.

[0153] During the initial configuration of a Slave Blade as a backup console server, it needs to be provided with an address for notification. This notification could be via SNMP or via email. The information that is sent in the notification is the IP address of the backup console server that has assumed the functionality of the console server or has detected the dysfunction of the console server.

[0154] **FIG. 6** outlines the overall flowchart that illustrates the functioning of the console server. The console server software continuously performs the configuration and the operation phases. The configuration phases are performed not as frequently as the operation phase. The configuration phase is involved in identifying the presence and status of the board. Since only the blade in Slot **1** of the chassis is capable of executing command bus command, the Server Blade in Slot **1** may be a good candidate for the console server function.

[0155] If the console server blade is in Slot **1**, then it can detect the presence of a Slave board via side band signaling over the command bus. The console server then runs the configuration phase querying each of the Slave Boards their status. The console server accepts one of the following three responses from each Blade that has been detected to be present:

> **[0156]** (i) Board present and functioning: This implies that Board is present and is capable of redirecting Console traffic; (ii) Board present and not participating: This implies that the Board is present and has chosen not to participate in the redirection of Console traffic; and (iii) Board present and not functioning: This implies that the Board is present and the console server is not receiving any responses back. Only in console server is Slot **1** capable of differentiating this from Board not present as it capable of using the Command bus to determine board presence.

[0157] If the console server does not hear a Server Blade, it assumes that the blade is not functional and performs the appropriate notification.

[0158] A Server Blade can be set to mirror or not to mirror its console information down the RS-485 bus. This is achieved via setting a bit in NVRAM.

[0159] The inter-chassis communication board (e.g. link board) is an optional feature that allows the cascading of multiple chassis such that a single console server can serve multiple chassis. The Inter-chassis communication Blade also enables the backup console server to live in any other chassis. A bi-directional daisy chain approach has been adopted to ensure that enables the detection of link failures. **FIG. 7** schematically illustrates a particular embodiment of the design.

[0160] Two types of message formats are available for communications between the console server and a particular computing device (e.g. Slave Blade), including command messages and Acknowledge messages. All of the messages may be inter- or intra-chassis. A scope bit **200** determines whether the message is inter- or intra-chassis. For inter-chassis commands, the two bytes that immediately follow the function field denote Chassis ID and Slot ID, respectively. **FIG. 8** illustrates the bit fields of command messages. The definition of each bit field is described in **FIG. 9**.

[0161] Receive command **220** is issued by the console server to send console data to the Slave Blade. This usually results in the Slave Blade receiving data from the console server. On successful receipt of the data, the Slave Blade responds using an Acknowledge command. The format that the console server uses to send data is to initially send two bytes that state the number of bytes of data that it intends to transmit, followed by the data.

[0162] Transmit command **222** is issued by the console server to request the Slave Blade to transmit data. This usually results in the Slave Blade sending data to the console server using the Acknowledge command. The format that the Slave Blade uses to convey data is to first send two bytes that state the number of bytes of data that the Slave Blade intends to transmit, followed by the data. If the Slave Blade has no data, it sends two bytes with all bits set to zero, indicating that it has no data to send.

[0163] Identify command **224** is issued by the Console Serve to request a Slave Blade to identify its presence. The typical response from the Slave Blade is an Acknowledge command.

[0164] Status command **226** is issued by the console server to request the Slave Blade to transmit its status. This typically results in the Slave Blade responding using the Acknowledge command to indicate whether or not it has any data available or other status information that may be relevant.

[0165] Re_sync command **228** is issued by the console server to request that all the Slave Blades resynchronize. This usually results in the Slave Blade performing an internal resynchronization operation. No response to the console server is expected to the Slave Blades.

[0166] Set command **230** is issued by the console server to request that the Slave Blade set certain fields in its internal registers. This usually results in the Slave Blade setting the bits and informing the console server via an Acknowledge command. If the Slave Blade is unable to set the registers, it responds back with the error bit set in the Acknowledge.

[0167] Get command is issued by the console server to request the Slave Blade to transmit certain fields in the internal registers of the Slave Blade, to the console server. This typically results in the Slave Blade sending the bits to the console server via an Acknowledge command. If the Slave Blade is unable to get the registers, it responds to the console server with an error bit set in the Acknowledge.

[0168] The bit fields of Acknowledge messages and their potential values and meanings are illustrated in **FIG. 10**. Acknowledge messages may be sent by the console server or a Slave Blade. The Slave Blade sends this message as a response to identify, transmit, status, receive, get or set commands that are sent by the console server. In the case where a Slave Blade is responding to a transmit, identify and get command, the Acknowledge message sent by the Slave Blade also contains data.

[0169] The console server also sends Acknowledge messages as a response to receiving data bytes from a Slave Blade. The Slave Blade sends this message as a response to a receive, status or set command sent by the console server. In response to such messages, only an Acknowledge command is sent. The Acknowledge message can be either a single or two byte long message depending on the information that needs to be conveyed. Acknowledge messages are intended to follow as a consequence of command messages. They are in all three types of Acknowledge messages.

[0170] Although the present invention has been described in several embodiments, a myriad of changes and modifications may be suggested to one skilled in the art, and it is intended that the present invention encompass such changes and modifications as fall within the scope of the present appended claims.

What is claimed is:

1. A computing device, comprising:

a console;

a console interface operable to transmit console information associated with the console;

a memory module operable to receive the console information; and

the memory module being further operable to store the console information for retrieval by an operator of the computing device.

2. The computing device of claim 1, wherein the memory module comprises a buffer.

3. The computing device of claim 1, wherein the memory module is operable to periodically transmit historical console information to a server coupled with the computing device.

4. The computing device of claim 3, wherein the server is operable to transmit periodic requests to the computing device to transmit the historical console information to the server.

5. The computing device of claim 4, wherein the requests comprise interrupt driven/on demand requests.

6. The computing device of claim 3, wherein the memory module is operable to transmit the historical console information to the server in response to an event.

7. The computing device of claim 3, wherein the memory module is operable to transmit the historical console information to the server at predetermined time intervals.

8. The computing device of claim 1, wherein the console information comprises real-time console information and the memory module is further operable to transmit real-time console information to a server coupled with the computing device.

9. The computing device of claim 1, wherein the memory module is further operable to transmit the console information to a server coupled with the computing device over a distributed communication network.

10. A system, comprising:

a first computing device, including a first console and a first console interface operable to transmit first console information associated with the first console;

a second computing device coupled for communication with the first computing device, the second computing device having a memory module operable to receive the first console information; and

the memory module being further operable store the first console information.

11. The system of claim 10, wherein the second computing device is further operable to provide first historical console information to an operator of the second computing device, wherein the first historical console information includes the stored first console information.

12. The system of claim 10, further comprising:

a third computing device, including a second console and second console interface operable to transmit second console information associated with the second console; and

the memory module being further operable to receive and store the second console information.

13. The system of claim 12, wherein the memory module is further operable to provide second historical console information to an operator of the second computing device, wherein the second historical console information includes the stored second console information.

14. The system of claim 10, wherein the memory module comprises a buffer.

15. The system of claim 10, wherein the second computing device is further operable to poll the first computing device periodically to request the transfer of at least a portion of the first console information.

16. The system of claim 10, wherein the first and second computing devices are coupled over a distributed communications network.

17. The system of claim 10, wherein the first computing device comprises a server processing card.

18. The system of claim 10, wherein the first and second computing devices are coupled for communication using an RS485 bus.

19. A method for storing console information, comprising:

transmitting console information associated with a console, from a console interface;

receiving the console information at a memory module; and

storing the console information at the memory module.

20. The method of claim 19, further comprising presenting historical console information to a graphical user interface in response to a request from a user, wherein the historical console information comprises the stored console information.

21. The method of claim 19, further comprising transmitting periodic requests to the console interface to transmit the console information to a computing device coupled for communication with the memory module.

22. The method of claim 19, further comprising transmitting the console information to a computing device coupled for communication with the memory module, at predetermined time intervals.

23. A method for storing console information, comprising coupling a first computing device and a second computing device, the first computing device including a first console and a first console interface, and the second computing device including a memory module;

transmitting first console information associated with the first console from the first console interface to the memory module;

receiving the first console information at the memory module; and

storing the first console information at the memory module.

24. The method of claim 23, further comprising providing first historical console information to an operator of the second computing device, wherein the first historical console information includes the stored first console information.

25. The method of claim 23, further comprising:

coupling a third computing device with the second computing device, the third computing device including a second console and a second console interface;

transmitting second console information associated with the second console from the second console interface to the memory module;

receiving the second console information at the memory module; and

storing the second console information at the memory module.

26. The method of claim 23, further comprising transmitting periodic requests from the second computing device to the first computing device, requesting the transfer of at least a portion of the first console information.

27. Logic encoded in media for storing console information, the logic operable to perform the following steps:

transmit console information associated with a console, from a console interface;

receive the console information at a memory module; and

store the console information at the memory module.

28. The logic encoded in media of claim 27, wherein the logic is further operable to present historical console information to a graphical user interface in response to a request from a user, wherein the historical console information comprises the stored console information.

29. The logic encoded in media of claim 27, wherein the logic is further operable to transmit periodic requests to the

console interface, to transmit the console information to a computing device, coupled for communication with the memory module.

30. The logic encoded in media of claim 27, wherein the logic is further operable to transmit the console information to a computing device coupled for communication with the memory module, at predetermined time intervals.

31. The logic encoded in media for storing console information associated with a first computing device which is coupled for communication with a second computing device, the first computing device computing a first console and a first console interface, and the second computing device including a memory module, the logic operable to perform the following steps:

transmit first console information associated with the first console from the first console interface to the memory module;

receive the first console information at the memory module; and

store the first console information at the memory module.

32. The logic encoded in media of claim 31, wherein the logic is further operable to provide first historical console information to an operator of the second computing device, wherein the first historical console information includes the stored first console information.

33. The logic encoded in media of claim 31, wherein a third computing device is coupled with the second computing device, the third computing device including a second console and a second console interface, the logic being further operable to:

transmit second console information associated with the second console from the second console interface to the memory module;

receive the second console information at the memory module; and

store the second console information at the memory module.

34. The logic encoded in media of claim 31, wherein the logic is further operable to transmit periodic requests from the second computing device to the first computing device, requesting the transfer of at least a portion of the first console information.

35. A system for storing console information, comprising:

means for transmitting console information associated with a console, from a console interface;

means for receiving the console information at a memory module; and

means for storing the console information at the memory module.

36. The system of claim 35, further comprising means for presenting historical console information to a graphical user interface in response to a request from a user, wherein the historical console information comprises the stored console information.

37. The system of claim 35, further comprising means for transmitting periodic requests to the console interface to transmit the console information to a computing device coupled for communication with the memory module.

38. The system of claim 35, further comprising means for transmitting the console information to a computing device coupled for communication with the memory module, at predetermined time intervals.

39. A system for storing console information, comprising:

means for coupling a first computing device and a second computing device, the first computing device including a first console and a first console interface, and the second computing device including a memory module;

means for transmitting first console information associated with the first console from the first console interface to the memory module;

means for receiving the first console information at a memory module; and

means for storing the first console information at the memory module.

40. The system of claim 39, further comprising means for providing first historical console information to an operator of the second computing device, wherein the first historical console information includes the stored first console information.

41. The system of claim 39, further comprising:

means for coupling a third computing device with the second computing device, the third computing device including a second console and a second console interface;

means for transmitting second console information associated with the second console from the second console interface to the memory module;

means for receiving the second console information at the memory module; and

means for storing the second console information at the memory module.

42. The system of claim 39, further comprising means for transmitting periodic requests from the second computing device to the first computing device, requesting the transfer of at least a portion of the first console information.

* * * * *